

Nr. 8/85 August

DM 6,50, sfr 6,50, öS 50, Lit 5900, hfl 7,50

# PEEKER

MAGAZIN FÜR APPLE-COMPUTER

Laufwerktestprogramm

Graf-quattro

Double-Lores-Erweiterungen

Adreßverwaltung mit dBASE

Wordstar und FX-80

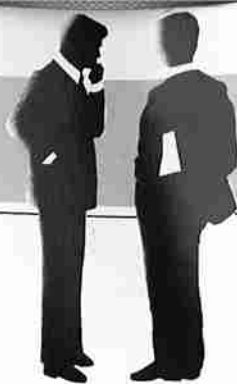
Fakultäten

Polaroid-System

Hüthig  
PUBLIKATION



Grafik  
auf dem Mac



**NEU!**  
Band 2!



## Apple ProDOS für Aufsteiger

Mit ausführlichen Programmbeispielen

von Ulrich Stiehl

Band 2: 1. Auflage 1985,  
208 S., kart., DM 30,—  
ISBN 3-7785-1036-3

Begleitdiskette  
DM 28,—

Der zweite Band von „ProDOS für Aufsteiger“ ist wieder ein typisches Tips- und Tricks-Buch, das viele nützliche Utilities und Hilfsroutinen enthält, die dem Applesoft- und Assemblerprogrammierer den Umgang mit ProDOS erleichtern sollen. Zunächst wird eine bewußt leichtverständliche Einführung in die Applesoft-Programmierung unter ProDOS gegeben, die sich insbesondere an diejenigen Leser wendet, die das alte DOS 3.3 nur noch „am Rande“ kennengelernt haben. Im Anschluß daran wird gezeigt, wie man einige Unzulänglichkeiten des BASIC.SYSTEMS durch Tricks beheben kann, z. B. Einlesen von Strings mit Komma und Doppelpunkt, Laden und Speichern von Zahlen als Binärdateien, Simulierung des fehlenden MON-Befehls u. a.

Den Hauptteil des Buches bilden sofort einsatzfähige Utilities, die alle klassischen „Pflichtübungen“ abdecken, z. B. Dateileseprogramme mit ASCII- und Hex-Dump, Dateikopierprogramme, Diskettenformatier- und -kopierprogramme, Diskettenvergleichsprogramme, Konvertierungsprogramme (DOS 3.3 nach ProDOS), Bad-Block-Programme u. a. Zur Anwendung dieser Utilities sind keinerlei Assemblerkenntnisse erforderlich.

**BESTELLCOUPON**

Buchtitel

Name

Straße

Ort

Unterschrift

Bitte ausfüllen und an Hüthig Vertriebs-  
service, Postfach 102869 · 6900 Hei-  
delberg schicken.



Das Gesetz der Sättigung hat jeder schon am eigenen Leibe verspürt. In der Psychologie sprechen wir von abnehmender Reizwirkung, in der Wirtschaftswissenschaft von Marktsättigung: Die erste Portion eines saftigen Steaks lassen wir uns einiges kosten; auch das zweite Tournedos ist vielen ihr Geld wert; aber nach dem dritten Stück haben wir sprichwörtlich „die Schnauze voll“ – die Sättigung ist erreicht.

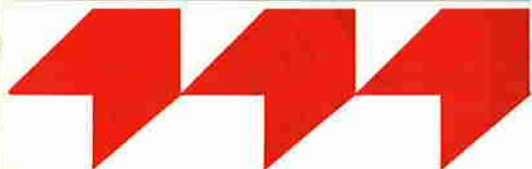
Was wir zur Zeit in der Mikrocomputerbranche erleben, ist sowohl eine psychologische als auch eine ökonomische Sättigung: psychologisch deshalb, weil der Reiz des Neuen verfliegen ist, ökonomisch deshalb, weil der Mikrocomputer inzwischen in vielen Büros und Wohnungen steht. Wie an den anderen Pionieren der Mikro-Ära ist auch an der Firma Apple diese „Sättigung“ nicht spurlos vorübergegangen: Man spricht bereits von weiteren Entlassungen – auch in Deutschland –, und Übernahmegerüchte halten sich hartnäckig in allen größeren Wirtschaftsblättern. Ein durchgreifendes Revirement in der Führungsetage in Cupertino hat mittlerweile sein zweites Opfer gefunden:

Steven Jobs wurde nahegelegt, vom Vorstand in den Aufsichtsrat überzuwechseln – ein probates Mittel, um unliebsame Manager aufs Abstellgleis zu schieben. Wie berichtet, hatte Steve Wozniak bereits vorher seinen Hut genommen. Damit sind die beiden legendären Gründer des Apple-Imperiums ins zweite Glied zurückgetreten.

Apple war die einzige Mikrocomputerfirma, bei der der Begriff des „Personal“-Computers noch eine andere, tiefere Bedeutung hatte, nämlich die Bedeutung des von einer Persönlichkeit geschaffenen und geprägten Mikrocomputers, wobei ich hier namentlich an Steve Wozniak als den technischen Kopf des Zweigespanns denke. Dies wird sich nun ändern, denn die Koppelung Apple/Sculley hat einen schaleren Sinn als die Assoziation Apple/Wozniak. Woz stand für Genie. Sculley steht für Pepsi. Nach den sieben fetten Jahren kommen jetzt die sieben anderen. Da ist kein Platz mehr für geniale Eigenbrötler – oder um mit G. B. Shaw zu sprechen: „Genius demanding bread is given a stone ...“

Ulrich Stiehl





# INHALT

# 8/85

#### Impressum

Peeker  
Magazin für Apple-Computer  
2. Jahrgang 1985  
ISSN 0176-9200  
© für den gesamten Inhalt  
einschließlich der Programme  
Dr. Alfred Hüthig Verlag,  
Heidelberg 1985

Verleger und Herausgeber:  
Dipl.-Kfm. Holger Hüthig  
Geschäftsführung Zeitschriften:  
Heinz Melcher  
Chefredakteur:  
Ulrich Stiehl (us) Tel. (06221) 48 93 52  
(Bitte nur in redaktionellen Angelegenheiten  
anrufen)

Anzeigenleitung:  
Jürgen Maurer, Tel. (06221) 48 92 18  
z. Zt. gilt Anzeigenpreisliste Nr. 3  
Vertriebsleitung:  
Ruth Biller, Tel. (06221) 48 92 80  
Produktionsleitung: Gunter Sokollek  
Gestaltung: Rainer Schmitt  
Titelbild: Creative Computer  
Service, Mannheim



# PEEKER

## MAGAZIN FÜR APPLE-COMPUTER

### Technik

Testprogramme für Apple-II-Diskettensysteme  
Analyse von Disketten, Laufwerk und Controller  
von Dipl.-Ing. Gerhard Berg 6

### ProDOS

ProDOS für Anfänger  
Teil 3: Geheimnisse von BSAVE und BLOAD  
von Ulrich Stiehl 18

### Grafik

Graf-quattro Teil 3: Tricks über Tricks  
von Nino G. Barbieri 24

Double-Lores-Applesoft-Erweiterungen  
von Karl-Walter Bott 32

### CP/M

Adreßverwaltungsprogramm mit dBASE II  
von Ing. (grad.) Ernst Fischer 40

Wordstar druckt internationale Zeichensätze  
Frei definierte Sonderzeichen auf dem FX-80 nutzen  
von Dipl.-Ing. H. A. Rohrbacher 45

### Pascal

Tips und Tricks in Pascal  
Teil 1: Die versteckte Prozedur „ldsearch“  
von Dieter Geiß 48

### Assembler

Assembler-Pseudo-Opcode-Referenztafel  
von Dr. Jürgen B. Kehrel 51

Fakultäten  
von Roland und Manfred Fietkau 56

### Mecki

Microsoft Basic leicht geMACHt  
Teil 4: Die Grafik  
von Pit Capitain 60

### Hobby

Grafik-Demonstrationen  
von Ralf Knoke 67

Zeichenjagd. Ein Einzeiler  
von Hans-Peter Lendle 69

### Testberichte

Die Polaroid-Foto-Systeme getestet  
von Thomas Bühner und Prof. Dr. Klaus Hausmann 71

Das Bildverarbeitungssystem MAGIC 74

Siemens entdeckt neue Primzahl 74

TRICARD-Multifunktionskarte für Apple IIe 74

Copy-Killer jetzt in deutsch 76

Microfloppy mit 2 × 1 MByte 76

ProDOS-Debugger BUGBYTER 76

Beagle Graphics getestet  
von Rolf W. Becker 77

### Errata

SUPERDUMP und Apple IIc 50

RAM.FRE 70

Diversi-DOS 2-C 68

Inserentenverzeichnis 78

# Testprogramm für Apple-II- Diskettensysteme

von Dipl.-Ing. Gerhard Berg

Analyse von Disketten,  
Laufwerk und Controller



Im Rahmen der Artikelserie über Diskettensysteme bringen wir in dieser Ausgabe ein Programm zum Testen des Apple-Diskettensystems. Das Programm ist ein unerlässliches Werkzeug für alle, die sich näher mit Floppy-Laufwerken beschäftigen.

Insbesondere wird das Programm für folgende Anwendungen gebraucht:

- Test aller Controller- und Laufwerk-Funktionen,
- Justage von Diskettenlaufwerken,
- Fehlersuche und Reparatur von Laufwerken und Controllern,
- Test von Disketten,
- Anpassung von Nicht-Apple-Laufwerken,
- Funktionsanalyse von Controller und Laufwerk.

Dazu verfügt das Programm über folgende Eigenschaften:

- einfache Bedienung durch Menüsteuerung,
- Fehlermeldungen bei Bedienungsfehlern,
- grafische Ausgabe zur besseren Anschaulichkeit,
- leichtes Oszillographieren durch spezielle Trigger-Signale und Betriebsarten,
- Laden des Programms von Diskette oder Kasette,
- Betrieb aller Controller, die bus-seitig Apple-kompatibel sind,
- Betrieb aller Disketten-Laufwerke bis zu 80 Spuren.

## 1. Eingabe und Start des Programms

Da das Programm die hochauflösende Grafik benutzt, muß dafür gesorgt werden, daß das Programm oberhalb der Grafik-Seite 1 (ab \$4000) im RAM liegt. Um dies zu erreichen, müssen als erstes die folgenden Befehle eingegeben werden:

```
POKE 104, 64
POKE 16384, 0
NEW
```

Mit dem Befehl POKE 104, 64 wird der Anfang des BASIC-Programms von \$0800 nach \$4000 verschoben. Wegen einer Eigenart des Applesoft-Interpreters muß das erste Byte des Programmspeicherbereichs zu Null gesetzt werden. Dies geschieht mit dem Befehl POKE 16384, 0. Mit dem Befehl NEW werden alle anderen

### DISKTEST.B

```

1  *Unterprogramme zum Diskettensystem-Testprogramm
2  *von Dipl.-Ing. Gerhard Berg - 2.3.1985
3  *
4      ORG  $9000
5  *
6  SLOT EQU  $FA      ;Slot-Nr. * $10
7  *
8  *Positionier-Variablen
9  ZIELSP EQU  SLOT+1 ;Zielspur
10 AKTSP EQU  ZIELSP+1 ;Aktuelle Spur
11 BERUHZ EQU  AKTSP+1 ;Beruhigungszeit in msec
12 SCHRITT EQU  BERUHZ+1 ;Schritt-Zähler
13 ALTSP EQU  SCHRITT+1 ;Alte (vorige) Spur
14 *
15 *Schreib-/Lese-Variablen
16 MUSTER EQU  SLOT+1 ;Flußwechsel-Muster
17 BYTEL EQU  MUSTER+1 ;Byte-Zähler
18 BYTEH EQU  BYTEL+1 ; (2'er Komplement)
19 FEHLERL EQU  BYTEH+1 ;Fehler-Zähler
20 FEHLERH EQU  FEHLERL+1
21 MODE EQU  FEHLERL ;Schreib-Modus
22 MODEA EQU  MODE ;absolute Adressierung!!
23 INDEX EQU  MODE ;Index-Zähler
24 *
25 *Interne I/O-Adressen
26 TASTDAT EQU  $C000 ;Tastatur-Daten-Register
27 LAUTSP EQU  $C030 ;Lautsprecher
28 TRIGGER EQU  $C040 ;Trigger (Utility Strobe)
29 *
30 *Disketten-Controller-Adressen
31 *Effektive Adresse = Basis-Adresse + Slot-Nr. * $10
32 CONTR EQU  $C080 ;Controller-Basis-Adresse
33 Q6AUS EQU  $C08C ;Q6 ausschalten
34 Q6EIN EQU  $C08D ;Q6 einschalten
35 Q7AUS EQU  $C08E ;Q7 ausschalten
36 Q7EIN EQU  $C08F ;Q7 einschalten
37 *
38 * Q7 Q6 Funktion
39 -----
40 * AUS AUS Lesen
41 * AUS EIN Schreibschutz prüfen und
42 * Controller initialisieren
43 * EIN AUS Schreiben
44 * EIN EIN Datenregister laden
45 *
46 *
47 *Lese-/Prüf-Routine
48 *
9000: A6 FA 49 LESEN LDX SLOT ;X = Slot-Nr. * $10
9002: A9 00 50 LDA #0
9004: 85 FE 51 STA FEHLERL ;Fehler-Zähler löschen
9006: 85 FF 52 STA FEHLERH
9008: A4 FC 53 LDY BYTEL ;Y = Byte-Zähler low Byte
900A: BD 8C C0 54 LESEN1 LDA Q6AUS,X ;Byte lesen
900D: 10 FB 55 BPL LESEN1 ;warten bis Byte komplett
900F: C5 FB 56 CMP MUSTER ;mit Muster vergleichen
9011: F0 06 57 BEQ LESEN2 ;beide gleich -> Sprung
9013: E6 FE 58 INC FEHLERL ;Fehler-Zähler erhöhen
9015: D0 02 59 BNE LESEN2
9017: E6 FF 60 INC FEHLERH
9019: C8 61 LESEN2 INY ;Byte-Zähler erhöhen
901A: D0 EE 62 BNE LESEN1 ; (2'er Komplement)
901C: E6 FD 63 INC BYTEH
901E: D0 EA 64 BNE LESEN1 ;Byte-Zähler = 0 -> Ende
9020: 60 65 RTS
66 *
67 *
68 *Schreib-Routine
69 *
70 *MODE: Bit 7 = x, Bit 6 = 0 -> dauernd schreiben
71 * Bit 7 = 0, Bit 6 = 1 -> 1 Spur schreiben
72 * Bit 7 = 1, Bit 6 = 1 -> 1 Spur + Index
73 *
9021: A6 FA 74 SCHREIB LDX SLOT ;X = Slot-Nr. * $10
9023: 24 FE 75 BIT MODE ;V Bit setzen = Bit 6
9025: A5 FB 76 LDA MUSTER ;A = Flußwechsellmuster
9027: DD 8D C0 77 CMP Q6EIN,X ;Controller initialisieren
902A: 9D 8F C0 78 STA Q7EIN,X ;Schreiben einschalten +
902D: DD 8C C0 79 CMP Q6AUS,X ;... erstes Byte ausgeben
9030: 08 80 PHP
9031: 28 81 PLP
9032: EA 82 NOP
9033: EA 83 NOP
9034: 08 84 SCHR0 PHP
9035: 28 85 PLP

```



Vektoren initialisiert, so daß ein neues Programm eingegeben werden kann.

Danach muß das gesamte BASIC-Programm eingetippt werden. Wenn die Eingabe der Kommentare eingespart werden soll, muß trotzdem zumindest die Zeilennummer und „REM“ eingegeben werden, da oft (insbesondere beim Anfang von Unterprogrammen) auf diese Zeilen gesprungen wird. Die Zeilen, die nur einen Doppelpunkt enthalten, dienen nur der besseren Übersichtlichkeit und können ohne weiteres weggelassen werden. Das Assemblerprogramm braucht nicht extra eingegeben zu werden, da es in Form von DATA-Befehlen im BASIC-Programm bereits enthalten ist. Dies wurde so eingerichtet, um ein einfaches Laden von Kassette zu ermöglichen, falls das Diskettensystem defekt ist. Nach dem Programmstart wird über eine Prüfsumme getestet, ob die „DATA“-Befehle, die das Assemblerprogramm enthalten, richtig eingegeben wurden.

Wer die Arbeit des Eintippens sparen will, kann das Programm auch auf der Peeker-Sammeldiskette beziehen.

Zur Anpassung an das verwendete Laufwerk können die folgenden Zeilen geändert werden:

– Zeile 10140 definiert die Spurzahl des Laufwerks (z.B. 35, 40, 77 oder 80). Der maximale Wert für AS ist 80.

– Zeile 10150 definiert die Anzahl der Schrittmotor-Phasen pro Spur. Dieser Wert ist normalerweise 2.

– Zeile 10160 definiert die Beruhigungszeit (s. Peeker 3/85, S. 20), die das Programm nach jeder Positionierung wartet, bevor es mit dem Schreiben oder Lesen anfängt. (Hinzu kommt noch die Ausführungszeit des BASIC-Programms.) BZ wird in Millisekunden angegeben und darf jeden Wert zwischen 1 und 255 haben. Für das Apple-Disketten-Laufwerk ist BZ normalerweise 25. Für andere Laufwerke muß die Beruhigungszeit den Herstellerangaben entnommen werden.

– Zeile 10170 definiert den Steckplatz (Slot), in dem normalerweise der zu testende Controller steckt. Der mit SN (1 bis 7) definierte Steckplatz wird beim Programmstart ausgewählt.

– Zeile 10180 definiert das normalerweise zu testende Laufwerk. Das mit LN definierte Laufwerk wird bei Programmstart ausgewählt. LN darf den Wert 1 oder 2 haben.

```

9036: C9 92 86 SCHR1  CMP  #92      ;12 microsec F.w.abstand?
9038: D0 02 87       BNE  SCHR2    ;nein -> Sprung
903A: E6 FF 88       INC  FEHLERH  ;4 microsec zusätzlich
903C: 9D 8D C0 89 SCHR2  STA  Q6EIN,X  ;Byte ausgeben
903F: DD 8C C0 90     CMP  Q6AUS,X
9042: 70 0A 91       BVS  SPSCHR   ;1 Spur -> Sprung
          92
          93 *
9044: EA 94         *dauernd schreiben, bis eine Taste gedrückt wird
          94       NOP
9045: AC 00 C0 95     LDY  TASTDAT  ;Taste gedrückt?
9048: 10 EA 96       BPL  SCHR0    ;nein -> weiter schreiben
904A: BD 8E C0 97 ENDSCHR LDA  Q7AUS,X  ;Schreiben ausschalten
904D: 60 98         RTS
          99
          100 *
904E: E6 FC 101     *1 Spur schreiben
          101 SFSCHR  INC  BYTEL   ;Byte-Zähler erhöhen
          102       BNE  SCHR0    ; (2'er Komplement)
9050: D0 E2 102     INC  BYTEL   ;Byte-Zähler <> 0 ->
9052: E6 FD 103     INC  BYTEL   ; weiter schreiben
9054: D0 E0 104     BNE  SCHR1    ; absolute Adressierung!!
9056: 2C FE 00 105   BIT:  MODEEA
9059: 10 EF 106     BPL  ENDSCHR  ;kein Index -> Ende
          107 *
          108 *Index für Umdrehungsdauermessung schreiben
905B: A0 03 109     INDSCHR LDY  #3      ;3 Index Bytes
905D: A9 AA 110     INDSCH1 LDA  #AA      ;Flußwechsellmuster 1010
905F: 9D 8D C0 111   STA  Q6EIN,X  ;Byte ausgeben
9062: DD 8C C0 112   CMP  Q6AUS,X
9065: 48 113       PHA
9066: 68 114       PLA
9067: 48 115       PHA
9068: 68 116       PLA
9069: 88 117       DEY      ;Byte-Zähler erniedrigen
906A: F0 DE 118     BEQ  ENDSCHR  ;Zähler = 0 -> Ende
906C: D0 EF 119     BNE  INDSCH1 ;Zähler > 0 -> weiter
          120 *
          121 *
          122 *Anzahl Bytes pro Spur messen (vom 1. zum 2. Index)
          123 *
906E: A6 FA 124     UMDR  LDX  SLOT    ;X = Slot-Nr. * $10
9070: A9 03 125     LDA  #3
          126     STA  INDEX    ;Index-Zähler setzen
9072: 85 FE 126     STA  INDEX    ;Index-Zähler erniedrigen
9074: C6 FE 127     UMDR1  DEC  INDEX    ;2. Index -> Sprung
9076: F0 24 128     BEQ  UMDR5    ;Y = Byte-Zähler low Byte
9078: A0 00 129     LDY  #0
907A: 84 FD 130     STY  BYTEH    ;Byte-Zähler löschen
          131 *
          132 *warten bis Index vorbei
907C: BD 8C C0 133   UMDR2  LDA  Q6AUS,X  ;Byte lesen
907F: 10 FB 134     BPL  UMDR2    ;warten bis Byte komplett
9081: C8 135       INY      ;Byte-Zähler erhöhen
9082: D0 04 136     BNE  UMDR3
9084: E6 FD 137     INC  BYTEH
9086: 30 14 138     BMI  UMDR5    ; > 32767 Bytes -> Fehler
9088: C9 FF 139     UMDR3  CMP  #92      ;4 microsec Abstand?
908A: D0 F0 140     BNE  UMDR2    ;nein = Index -> Sprung
          141 *
          142 *warten auf Index
908C: BD 8C C0 143   UMDR4  LDA  Q6AUS,X  ;Byte lesen
908F: 10 FB 144     BPL  UMDR4    ;warten bis Byte komplett
9091: C9 AA 145     CMP  #AA      ;Index?
9093: F0 DF 146     BEQ  UMDR1    ;ja -> Sprung
9095: C8 147       INY      ;Byte-Zähler erhöhen
9096: D0 F4 148     BNE  UMDR4
9098: E6 FD 149     INC  BYTEH
909A: 10 F0 150     BPL  UMDR4    ; > 32767 Bytes -> Fehler
909C: 84 FC 151     UMDR5  STY  BYTEL    ;Byte-Zähler low Byte
909E: 60 152       RTN      RTS
          153 *
          154 *
          155 *Trigger für Spur-Justage mit CE-Disk erzeugen
          156 *BERUHZ = Mindest-Zeit bis zum nächsten Trigger in msec
          157 *
909F: A6 FA 158     SPTRIG  LDX  SLOT    ;X = Slot-Nr. * $10
90A1: BD 8C C0 159   SPTRIG1 LDA  Q6AUS,X  ;Byte lesen
90A4: 10 FB 160     BPL  SPTRIG1 ;warten bis Byte komplett
90A6: 2C 00 C0 161   BIT  TASTDAT  ;Taste gedrückt?
90A9: 30 F3 162     BMI  RTN      ;ja -> Abbruch
90AB: C9 FF 163     CMP  #9FF     ;Orientierungsburst?
90AD: D0 F2 164     BNE  SPTRIG1 ;nein -> auf Burst warten
90AF: 2C 30 C0 165   BIT  LAUTSP   ;Lautsprecher ansteuern
90B2: 2C 40 C0 166   BIT  TRIGGER  ;Trigger ansteuern
90B5: 20 08 91 167   JSR  BERUH   ;warten
90B8: F0 E5 168     BEQ  SPTRIG  ;immer springen
          169 *
          170 *
          171 *Positionier-Routine
          172 *
90BA: A9 00 173     POSIT  LDA  #0

```

```

90BC: 85 FE 174 STA SCHRITT ;Schritt-Zähler löschen
90BE: A5 FC 175 POSIT0 LDA AKTSP ;aktuelle Spur laden
90C0: 85 FF 176 STA ALTSP ;... und retten
90C2: 38 177 SEC ;Zielspur subtrahieren
90C3: E5 FB 178 SBC ZIELSP ;ergibt Spurdifferenz in A
90C5: F0 31 179 BEQ AMZIEL ;Spurdiff. = 0 -> am Ziel
90C7: B0 06 180 BCS POSIT1 ;Diff. > 0 - nach außen
181 *
182 *aktuelle Spur < Zielspur -> nach innen positionieren
90C9: 49 FF 183 BOR #$FF ;Absolutwert der Spurdiff.
90CB: E6 FC 184 INC AKTSP ;aktuelle Spur erhöhen
90CD: 90 04 185 BCC POSIT2
186 *
187 *aktuelle Spur > Zielspur -> nach außen positionieren
90CF: 69 FE 188 POSIT1 ADC #$FE ;l subtrahieren
90D1: C6 FC 189 DEC AKTSP ;aktuelle Spur erniedrigen
190 *
191 *Spurdifferenz in A mit Schritt-Zähler vergleichen
90D3: C5 FE 192 POSIT2 CMP SCHRITT
90D5: 90 02 193 BCC POSIT3 ;Spurdiff. < Schrittzahl
194 * -> mit Spurdifferenz Rampe runterlaufen
90D7: A5 FE 195 LDA SCHRITT ;Spurdiff. > Schrittzahl
196 * -> mit Schritt-Zähler Rampe hochlaufen
197 *
198 *wenn Rampe (Y) >= 12 würde -> Y nicht mehr ändern
90D9: C9 0C 199 POSIT3 CMP #12
90DB: B0 01 200 BCS POSIT4 ;A >= 12 -> Y nicht ändern
90DD: A8 201 TAY ;A < 12 -> A nach Y laden
90DE: 38 202 POSIT4 SEC ;C = 1 ->
90DF: 20 FC 90 203 JSR PHASE1 ; aktuelle Phase ein
90E2: B9 1F 91 204 LDA ZPHEIN,Y ;Einschaltzeit warten
90E5: 20 13 91 205 JSR WARTE
90E8: A5 FF 206 LDA ALTSP ;vorhergehende Phase ...
90EA: 18 207 CLC ;... mit C = 0 ...
90EB: 20 FE 90 208 JSR PHASE2 ;... ausschalten
90EE: B9 2B 91 209 LDA ZPHAUS,Y ;Ausschaltzeit warten
90F1: 20 13 91 210 JSR WARTE
90F4: E6 FE 211 INC SCHRITT ;Schritt-Zähler erhöhen
90F6: D0 C6 212 BNE POSIT0
213 *
90F8: 20 08 91 214 AMZIEL JSR BERUH ;Beruhigungszeit warten
90FB: 18 215 CLC ;aktuelle Phase aus
216 *
217 *Schrittmotor Phase ein- bzw. ausschalten
90FC: A5 FC 218 PHASE1 LDA AKTSP ;aktuelle Spur laden
90FE: 29 03 219 PHASE2 AND #3 ;Phasen # maskieren und
9100: 2A 220 ROL ;... nach links schieben
221 * C (Phase aus/ein) ist jetzt Bit 0
9101: 05 FA 222 ORA SLOT ;Slot-Nr. * $10 addieren
9103: AA 223 TAX
9104: BD 80 C0 224 LDA CONTR,X ;Phase ein-/ausschalten
9107: 60 225 RTS
226 *
227 *Beruhigungszeit warten (BERUHZ * 1 msec)
9108: A4 FD 228 BERUH LDY BERUHZ
910A: A9 0A 229 BERUH1 LDA #10 ;10 * 100 microsec
910C: 20 13 91 230 JSR WARTE ; = 1 msec warten
910F: 88 231 DEY
9110: D0 F8 232 BNE BERUH1
9112: 60 233 RTS
234 *
235 *Warte-Routine (verzögert um A * 100 microsec)
9113: A2 12 236 WARTE LDX #18
9115: CA 237 WARTE1 DEX
9116: D0 FD 238 BNE WARTE1
9118: EA 239 NOP
9119: 38 240 SEC
911A: E9 01 241 SBC #1 ;A um 1 erniedrigen
911C: D0 F5 242 BNE WARTE
911E: 60 243 RTS
244 *
245 *Zeittabellen für Rampe: Phase ein / Phase aus
911F: 01 30 28 246 ZPHEIN HEX 01302824201E1D1C1C1C1C
9122: 24 20 1E 1D 1C 1C 1C 1C
912A: 1C
912B: 70 2C 26 247 ZPHAUS HEX 702C26221F1E1D1C1C1C1C
912E: 22 1F 1E 1D 1C 1C 1C 1C
9136: 1C

```

311 Bytes

Nachdem das Programm vollständig eingegeben ist, wird es mit „SAVE DISKTEST“ auf Diskette und/oder „SAVE“ auf Kassette gespeichert. Mit Hilfe eines Texteditors sollte zusätzlich die Text-Datei **DISKTEST.START** auf Diskette gespeichert werden. Damit kann das Programm einfach durch Eingabe von „EXEC DISKTEST.START“ geladen und gestartet werden.

Zum Laden und Starten von Kassette werden die folgenden Befehle benötigt:  
POKE 104,64; POKE 16384,0; LOAD; RUN

## 2. Bedienung des Programms

Nach dem Start des Programms wird auf dem Bildschirm das Hauptmenü (s. **Tabelle 1**) mit allen Befehlen des Testprogramms ausgegeben. Die Befehle werden durch Eingabe des zugehörigen Buchstabens ausgeführt, die in Klein- oder Großschreibung eingegeben werden können.

**C** – Mit dem C-Befehl wird der Steckplatz (Slot) ausgewählt, in dem der zu testende Controller steckt. Nach Eingabe des „C“ springt der Cursor ans Ende der Befehlszeile. Nachdem die Steckplatz-Nummer (1 bis 7 – ohne RETURN) eingegeben worden ist, prüft das Programm, ob sich in dem angegebenen Steckplatz tatsächlich ein Disketten-Controller befindet. Wenn nicht, wird eine Fehlermeldung ausgegeben. Wenn sich in dem Steckplatz ein Disketten-Controller befindet, wird aus dem Inhalt des PROMs eine Prüfsumme gebildet. Falls die Prüfsumme nicht stimmt, wird ebenfalls eine Fehlermeldung ausgegeben. Bei Controllern mit abweichendem PROM kann die Fehlermeldung ignoriert oder das Programm für die andere Prüfsumme entsprechend abgeändert werden (Zeile 30080).

**D** – Mit dem D-Befehl wird das zu testende Laufwerk (**Drive**) ausgewählt. Jedesmal, wenn die Taste „D“ gedrückt wird, wird zwischen Laufwerk 1 und 2 hin- und hergeschaltet. Das ausgewählte Laufwerk wird am Ende der Zeile angegeben.

**E** – Mit dem E-Befehl wird das ausgewählte Laufwerk ein- bzw. ausgeschaltet. Ist das Laufwerk ausgeschaltet, so wird es durch Drücken der Taste „E“ eingeschaltet. Wird die Taste „E“ das nächste Mal betätigt, so wird das Laufwerk wieder ausgeschaltet. Am Ende der Zeile ist angegeben, ob das Laufwerk momentan ein- oder ausgeschaltet ist.

**O** – Der O-Befehl  $\emptyset$  dient zur Positionierung des Laufwerks auf die Spur **O**. Nachdem ein Laufwerk das erste Mal eingeschaltet wurde, muß es zunächst auf Spur 0 positioniert werden, bevor weitere Befehle ausgeführt werden können.

**I** – Mit dem I-Befehl wird das ausgewählte Laufwerk von der momentanen Position um jeweils eine Spur nach **Innen** positioniert. Die jeweilige Spurposition wird am Ende des P-Befehls ausgegeben.

**A** – Der A-Befehl positioniert das ausgewählte Laufwerk um eine Spur nach **außen**.

**P** – Mit Hilfe des P-Befehls kann direkt auf eine beliebige Spur **positioniert** werden. Nach Drücken der Taste „P“ springt der Cursor an das Ende der Befehlszeile. Nachdem die gewünschte Spurnummer und RETURN eingegeben wurde, positioniert das Laufwerk auf diese Spur.

**H** – Mit dem H-Befehl positioniert das Laufwerk ständig zwischen zwei beliebigen Spuren **hin und her**. Damit kann der Positioniermechanismus über längere Zeit getestet werden. Nach Drücken der Taste „H“ müssen zunächst die beiden Spuren eingegeben werden (jeweils mit RETURN abgeschlossen). Danach startet das Positionieren, das jederzeit durch Drücken einer beliebigen Taste abgebrochen werden kann. Zum leichteren Oszillographieren wird nach jedem zweiten Positionieren ein Trigger-Impuls am Utility-Strobe-Ausgang (Game I/O-Stecker Stift 5) erzeugt.

**T** – Der T-Befehl **testet**, ob die Diskette in dem ausgewählten Laufwerk schreibgeschützt ist oder nicht.

**F** – Mit dem F-Befehl wird der beim Schreiben und Lesen verwendete **Flußwechselabstand** ausgewählt. Zur Wahl stehen drei Flußwechsellmuster mit den drei beim Apple-Aufzeichnungsverfahren vorkommenden Flußwechselabständen von 4, 8 und 12 Mikrosekunden. Jedesmal, wenn die Taste „F“ gedrückt wird, wird auf das nächste Flußwechsellmuster weitergeschaltet.

**M** – Mit dem M-Befehl wird die Betriebsart (**Modus**) für Schreiben und Lesen ausgewählt. Zur Wahl stehen: „dauernd“, „1 Spur“ und „ganze Diskette“. Jedesmal, wenn die Taste „M“ gedrückt wird, wird die nächste Betriebsart für Schreiben und Lesen vorgewählt.

**L** – Der L-Befehl startet das **Lesen** von der Diskette. Nach jedem Lesevorgang wird die Anzahl der gelesenen und der davon fehlerhaften Disk-Bytes ausgegeben. Voraussetzung ist natürlich, daß die Diskette vorher mit dem gleichen Flußwechsellmuster beschrieben wurde. Je nach ausgewählter Betriebsart hat der L-Befehl unterschiedliche Funktionen:

– Das Lesen in der Betriebsart „dauernd“ bewirkt, daß das Lesen der momentanen Spur ständig wiederholt wird, bis es durch Drücken einer beliebigen Taste abgebrochen wird. Nach jedem Lesevorgang wird das Ergebnis auf dem Bildschirm ausgegeben.

– In der Betriebsart „1 Spur“ wird die momentane Spur einmal gelesen.

– In der Betriebsart „ganze Diskette“ werden alle Spuren der Diskette nacheinander einmal gelesen und das Ergebnis für alle Spuren auf dem Bildschirm ausgegeben. Nachdem alle Spuren gelesen wurden oder das Lesen durch Drücken einer beliebigen Taste abgebrochen wurde, erscheint ein Untermenü, das folgende Funktionen zuläßt:

– T – Der T-Befehl bewirkt die Ausgabe der Testergebnisse in Textform auf dem Bildschirm. Dies ist die gleiche Form, in der die Ergebnisse auch während des Tests selbst ausgegeben werden.

– D – Der D-Befehl bewirkt die Ausgabe der Testergebnisse auf einem Drucker. Das Programm geht davon aus, daß sich das Druckerinterface in Steckplatz (Slot) 1 befindet. Ist dies nicht der Fall, muß Zeile 25610 im BASIC-Programm entsprechend geändert werden.

– G – Durch den G-Befehl werden die Testergebnisse grafisch in Form eines Balkendiagramms auf dem Bildschirm ausgegeben. Auf der waagerechten Achse werden die Spuren aufgetragen und auf der senkrechten Achse im logarithmischen Maßstab die Anzahl der Lesefehler. Striche am linken Bildrand markieren die Werte für 0, 10, 100 und 1000 Lesefehler.

– H – Durch Drücken der Taste „H“ gelangt man zurück zum Hauptmenü.

**S** – Der S-Befehl des Hauptmenüs startet das **Schreiben** auf die Diskette. Zuvor wird geprüft, daß die Diskette nicht schreibgeschützt ist, und der Bediener wird gewarnt, welche Spuren überschrieben werden. Zur Bestätigung, daß wirklich geschrieben werden soll, muß ein „J“ (Ja) eingegeben werden.

– In der Betriebsart „dauernd“ wird ununterbrochen geschrieben, bis eine beliebige Taste gedrückt wird. Diese Betriebsart ist speziell zum Oszillographieren des Schreibvorgangs vorgesehen.

– In der Betriebsart „1 Spur“ wird die momentane Spur einmal mit dem ausgewählten Flußwechsellmuster beschrieben.

– In der Betriebsart „ganze Diskette“ werden alle Spuren einmal beschrieben.

**W** – Der W-Befehl ermöglicht **abwechselndes Schreiben und Lesen**. Die Funktion in den einzelnen Betriebsarten ist gleich dem L-Befehl, außer daß unmittelbar vor jedem Lesen die momentane Spur einmal beschrieben wird.

**U** – Der U-Befehl dient zur Messung und Justage der **Umdrehungsdauer** (Drehzahl). Die Abweichung von der nominalen Umdrehungsdauer wird ständig numerisch und grafisch ausgegeben, bis eine beliebige Taste gedrückt wird. Positive Abweichungen bedeuten, daß die Umdrehungsdauer zu hoch ist bzw. daß sich die Diskette zu langsam dreht. Die Abweichung soll zwischen +1 % und –1 % liegen. Die optimale Drehzahl liegt bei etwa +0,5 % Abweichung.

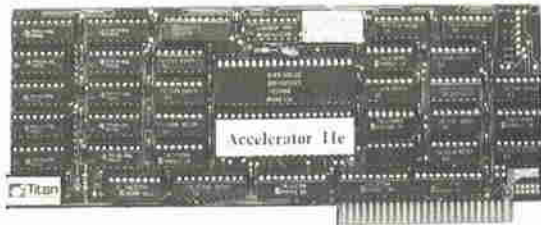
**J** – Der J-Befehl dient als Hilfe beim Überprüfen und **Justieren** der Spurlage mit Hilfe einer Justage-Diskette (CE-Disk) von BASF. Normalerweise braucht man beim Arbeiten mit dieser Diskette den Index zum Triggern des Oszillographen. Beim Apple-Laufwerk, das ja keine Index-Abfrage hat, kann man dann nur auf das Lesesignal triggern, was sehr schwierig ist. Das Programm schafft hier Abhilfe, indem es einen künstlichen Index generiert. Nach Eingabe des J-Befehls wird pro Umdrehung ein Impuls am Utility-Strobe-Ausgang (Game I/O Stecker Stift 5) generiert. Zusätzlich wird pro Umdrehung auch ein „Klick“ des Lautsprechers erzeugt.

Vor Ausgabe des Trigger-Signales wird noch überprüft, ob sich das Laufwerk auf der für die Justage richtigen Spur befindet. Bei 5,25-Zoll-Laufwerken ist dies bei einfacher Spurdichte Spur 16 und bei doppelter Spurdichte Spur 32. Befindet sich das Laufwerk nicht auf der richtigen Spur, wird eine entsprechende Fehlermeldung ausgegeben.

Die Kontrolle und Justage der Spurlage mit Hilfe der BASF-CE-Disk wird ausführlich in einem späteren Artikel erläutert.



# Accelerator™ IIe macht Ihren Apple® II, II Plus oder IIe dreieinhalbmal schneller.



Jetzt laufen VisiCalc®, Apple Writer, PASCAL, BASIC, Datenbanken usw. endlich ohne langen Zeitverlust.

Stecken Sie einfach die ACCELERATOR IIe Karte in irgendeinen Slot und beobachten Sie, wie Ihr Apple loslegt!

ACCELERATOR IIe besitzt seinen eigenen schnellen 6502 Prozessor und 80 K-Byte Hochgeschwindigkeitspeicher, einschließlich einer eingebauten schnellen Sprachkarte und schnellem RAM-Speicherplatz für die ROM-Sprache.

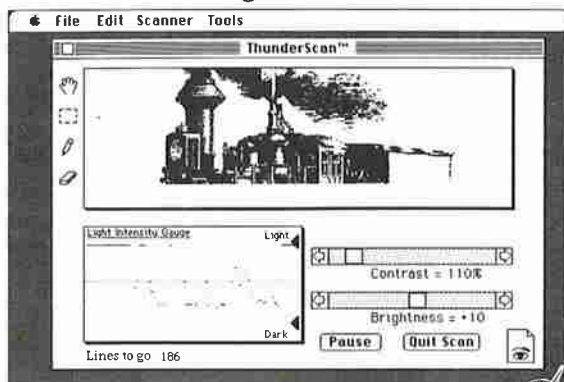
Direkt von Pandasoftware (Titan Distributor für Deutschland) oder bei Ihrem Applehändler.

## pandasoft Dr.-Ing. Eden

Uhlandstraße 195 · 1000 Berlin 12 · Mo-Fr 10-18 Uhr, Sa 10-13 Uhr  
Telefon: 0 30/31 04 23 · Telex: 1 85 859

# ThunderScan™

Ein neues optisches Lesegerät, das beliebige Vorlagen in MacPaint überträgt: Fotos, Zeichnungen, Landkarten und Illustrationen werden in den Apple-Imagewriter eingespannt und von einem Lesekopf, der das Farbband ersetzt, abgetastet.



- 32 Graustufen
- 80 Punkte/cm Auflösung
- Übertragungsmaßstab 25% - 400%
- Vorlagen bis 20 x 25 cm
- Nachträgliche Veränderung des Kontrasts und der Helligkeit.



ThunderScan

## pandasoft Dr.-Ing. Eden

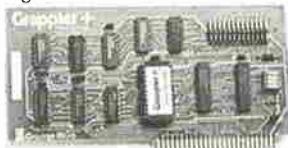
Uhlandstraße 195 · 1000 Berlin 12 · Mo-Fr 10-18 Uhr, Sa 10-13 Uhr  
Telefon: 0 30/31 04 23 · Telex: 1 85 859



**Druckerinterfaces** für Apple II+/e/ c/III Interfaces auf dem **neuesten Stand der Technik. Kompatibel** mit allen gängigen Druckern wie: APPLE, EPSON, STAR, NEC, OKIDATA usw. Passende Treiber-Software wird über Dip-Switch ausgewählt.



**Grappler+** **Grafikfähiges Druckerinterface** das keine Wünsche mehr offen läßt, ermöglichten die volle Kontrolle über **2 Dutzend Kommandos** über alle Möglichkeiten Ihres Druckers. Jetzt auch mit **IIe Features: Double Hires Grafics** und **80 Zeichen Dump** mittels Druckerpuffer nachrüstbar über Bufferboard.



Besitzt alle Vorzüge des Grappler+, hat aber zusätzlich einen integrierten **16 K Druckpuffer**, der auf **32 oder 64 K aufrüstbar** ist.



**Seriellles Druckerinterface** speziell für den **Apple Imagewriter**.



Seriell-nach-Parallel-Wandler für den IIc im Kabel integriert.



wie Hotlink, jedoch zusätzlich Imagewriter Emulation und Grafik Software-Diskette.

## pandasoft Dr.-Ing. Eden

Uhlandstraße 195 · 1000 Berlin 12 · Mo-Fr 10-18 Uhr, Sa 10-13 Uhr  
Telefon: 0 30/31 04 23 · Telex: 1 85 859

## Sie haben einen Apple...

### wir haben die Software...



### und die Hardware...



### wir haben die Bücher...



### und die Zeitschriften...



**\*Fordern Sie unseren Gratiskatalog an!**

ALLES FÜR DEN APPLE II+, IIe, IIc UND MACINTOSH

## pandasoft Dr.-Ing. Eden

UHLANDSTR. 195 · D-1000 BERLIN 12  
TEL.: (030) 310 423 · TELEX: 18 58 59

Autorenleiste: Apple Fachhändler MICROSOFT-Distributor

Ich besitze einen Apple. Bitte schicken Sie mir Ihren kostenlosen Katalog.  
Name: \_\_\_\_\_ Adresse: \_\_\_\_\_

**B** – Mit Hilfe des B-Befehls erfolgt die Rückkehr nach **BASIC**. Soll das Programm anschließend weiterbenutzt werden, so ist ein Warmstart mit „GOTO 10000“ möglich. Zu beachten ist, daß nach Verlassen des Programms keine DOS-Befehle mehr funktionieren, da das DOS aus Sicherheitsgründen „abgekoppelt“ wird. Soll mit dem Diskettensystem weitergearbeitet werden, so muß das DOS neu gebootet werden.

### 3. Aufbau und Funktion des Programms

Das Testprogramm besteht aus einem Applesoft-Programm für die Tastatur-Eingabe und Bildschirm-Ausgabe sowie einigen Assembler-Unterprogrammen für alle zeitkritischen Disketten-Operationen. Das BASIC-Programm liegt ab \$4000 und die Assembler-Unterprogramme ab \$9000 im RAM. Das BASIC-Hauptprogramm besteht aus den Zeilen 10000 bis 10630. Die Zeilen 11000 bis 29240 enthalten die Unterprogramme für die einzelnen Befehle und die Zeilen 30000 bis 49090 solche Unterprogramme, die von mehreren Befehlen gemeinsam benutzt werden. In den Zeilen 50000 bis 55520 erfolgt die Initialisierung des Programms.

Eine ausführliche Beschreibung des Programmablaufs ist aus Platzgründen nicht möglich. Es sollen deshalb nur einige Erläuterungen zur Ansteuerung des Controllers und Laufwerks gegeben werden.

Das Ein- bzw. Ausschalten des Laufwerks und das Umschalten zwischen den beiden Laufwerken erfolgt durch „PEEK“-Befehle unmittelbar vom BASIC-Programm aus (Zeile 33020 und 33030). Ebenso erfolgt die Abfrage des Schreibschutzes durch „PEEK“-Befehle (Zeile 36010 und 36020). Alle anderen Funktionen sind zeitkritisch und können deshalb nicht direkt im BASIC-Programm ausgeführt werden, sondern rufen über „CALL“-Befehle Assembler-Unterprogramme auf. Für alle Assembler-Unterprogramme wird in Adresse 250 (\$00FA) die Nummer des Steckplatzes (Slot) \* 16 übergeben (Zeile 30090). Zum **Positionieren** wird in Adresse 251 bis 253 (\$00FB bis \$00FD) die Zielspur, die momentane Spur und die Beruhigungszeit übergeben (Zeile 34020 und 34030). Die Positionier-Routine ist im Assembler-Programm in Zeile 171 bis 247 enthalten. Um die Laufwerke beim Positionieren unter gleichen Bedingungen wie im echten Betrieb zu testen, wurde der Positionier-Algorithmus unverändert von DOS 3.3 übernommen.

Zum Schreiben wird in Adresse 251 bis 254 (\$00FB bis \$00FE) das Flußwechselformat, die Anzahl der zu schreibenden Disk-Bytes (im 2er-Komplement) und der Schreibmodus übergeben (Zeile 38030 und 37020). Die Schreib-Routine ist im Assembler-Programm in den Zeilen 68 bis 119 enthalten.

#### a. Die Flußwechselformate

Das Testprogramm erlaubt die Auswahl zwischen drei verschiedenen Flußwechselformaten, die den drei beim Apple-Aufzeichnungsverfahren vorkommenden Frequenzen bzw. Flußwechselabständen entsprechen. Beim Apple-Aufzeichnungsverfahren ist eine Bitzelle 4 µsec lang. Die Definition der Datenflußwechsel ist die gleiche wie bei FM (s. Peeker 3/85, S. 19/20). Im Gegensatz zu FM und MFM werden beim Apple jedoch überhaupt keine Taktflußwechsel geschrieben. Die Bytes, die auf die Diskette geschrieben werden, werden Disk-Bytes genannt, da sie nicht direkt den eigentlichen Daten-Bytes entsprechen. Um eine kontinuierliche Flußwechselfolge mit 4 µsec Abstand zu schreiben, muß man demnach ein Disk-Byte mit dem Bitmuster 11111111 = \$FF = 255 verwenden. Das Bitmuster 10101010 = \$AA = 170 erzeugt eine kontinuierliche Aufzeichnung mit 8 µsec Flußwechselabstand. Etwas schwieriger wird es bei 12 µsec Abstand, was einem Bitmuster 100... entspricht, das sich nicht in einem 8-Bit-Disk-Byte unterbringen läßt. Mit einem einfachen Trick kann man aber mit dem Controller 9- oder 10-Bit-Disk-Bytes schreiben. Dazu werden die ersten 8 Bits in den Controller ausgegeben und der Controller fügt solange Nullen hinzu, bis das nächste Disk-Byte ausgegeben wird. Das Bitmuster 10010010 = \$52 = 146 erzeugt die Aufzeichnung 100100100 und damit das gewünschte Muster, wenn die Disk-Bytes im Abstand von 36 µsec (anstelle 32 µsec) an den Controller ausgegeben werden. **Bild 1** bis **3** zeigt die Aufzeichnung für die drei Flußwechselformate.

#### b. Die Spurkapazität

Die nominale Spurkapazität einer Apple-Diskette berechnet sich aus folgender Formel:

$$K_s = U : B$$

In der Formel ist  $K_s$  die (unformatierte) Kapazität pro Spur in Disk-Bits,  $U$  ist die Umdrehungsdauer der Diskette und  $B$  ist die Zeitdauer einer Bitzelle.

Bei nominaler Drehzahl ist die Umdre-

hungsdauer  $U = 200 \text{ msec} = 200\,000 \text{ µsec}$ .

Der Original-US-Apple hat eine Oszillatorfrequenz von  $f = 14,31818 \text{ MHz}$ , was einer Periodendauer von  $T = 1 : f = 69,841 \text{ nsec}$  entspricht. Der Prozessortakt wird durch 14fache Teilung erzeugt und hat somit eine Periodendauer von  $14 \cdot T = 978 \text{ nsec}$ . Jeder 65. Prozessortakt wird jedoch um 2 Oszillatorperioden verlängert und ist somit  $16 \cdot T = 1117 \text{ nsec}$  lang. Damit beträgt die mittlere Dauer eines Prozessortaktes  $(64 \cdot 14 \cdot T + 16 \cdot T) : 65 = 912 \cdot T : 65 = 980 \text{ nsec}$ . Die Länge einer Bitzelle ist schließlich 4 Prozessortakte und damit im Mittel  $4 \cdot 912 \cdot T : 65 = 3648 \cdot T : 65 = 3919,7 \text{ nsec} = 3,9197 \text{ µsec}$ .

Damit beträgt die Kapazität einer Spur  $K_s = 200\,000 : 3,9197 = 51024 \text{ Disk-Bits}$ . Für 8-Bit-Disk-Bytes ist damit die Kapazität pro Spur  $51024 : 8 = 6378 \text{ Disk-Bytes}$  und für 9-Bit-Disk-Bytes  $51024 : 9 = 5669 \text{ Disk-Bytes}$ .

#### c. Das Schreiben

Der Schreibmodus wird durch Bit 7 und 6 in Adresse 254 (\$00FE) definiert. Ein Wert von 64 (\$40) bedeutet, daß eine ganze Spur geschrieben werden soll. Um sicherzustellen, daß auch bei Drehzahlschwankungen eine vollständige, in sich geschlossene Spur entsteht, werden 10 % mehr Disk-Bytes geschrieben. An der Stelle, wo das Schreiben ausgeschaltet wird, entsteht eine sog. Stoßstelle. **Bild 4** zeigt ein Beispiel, wie eine solche Stoßstelle entsteht. Der Abstand  $t$  zwischen zwei gleichen Flußwechseln kann im Beispiel – je nach momentaner Umdrehungsdauer – jeden Wert zwischen 0 und 16 µsec haben. Je nachdem, welchen Wert  $t$  hat, tritt beim Lesen der Stoßstelle ein Fehler auf oder nicht.

Ein Wert von 0 für den Schreibmodus bedeutet, daß ununterbrochen geschrieben werden soll, bis eine Taste gedrückt wird.

#### d. Das Lesen

Beim Lesen werden in Adresse 251 bis 253 (\$00FB bis \$00FD) das Flußwechselformat und die Anzahl der zu lesenden Disk-Bytes übergeben (Zeile 38030). Die Anzahl der zu lesenden Disk-Bytes ist genau gleich der nominalen Spurkapazität. Das Assembler-Unterprogramm zum Lesen befindet sich in den Zeilen 47 bis 65. Bei Rückkehr vom Unterprogramm wird in Adresse 254 und 255 (\$00FE und \$00FF) die Anzahl der fehlerhaft gelesenen Disk-Bytes übergeben. Um die Software zu ver-



einfachen, wird beim Lesen nicht auf den Spuranfang synchronisiert, sondern das Lesen beginnt da, wo sich der Schreib-/ Lesekopf gerade befindet. Das Programm liest also auch die beim Abschalten des Schreibens entstandene Stoßstelle, was häufig ein fehlerhaftes Disk-Byte zur Folge hat. Aus diesem Grund ist ein Ergebnis mit *einem* Lesefehler praktisch als fehlerfrei anzusehen und wird in der grafischen Darstellung auch so ausgegeben.

### e. Die Umdrehungsdauer

Die Messung der Umdrehungsdauer erfolgt indirekt über eine Messung der Spurkapazität. Für ein einwandfreies Funktionieren der Software (insbesondere von Nibble-Kopierern) ist es erforderlich, daß

mindestens die nominale Spurkapazität auf die Diskette paßt. Für alle Euro-Apples, die mit einer von US-Apples etwas abweichenden Oszillatorfrequenz arbeiten, bedeutet das, daß die nominale Umdrehungsdauer nicht 200 msec beträgt, sondern dann erreicht ist, wenn 6378 Disk-Bytes auf eine Spur passen. Deshalb darf die Umdrehungsdauer auch nicht – falls vorhanden – mit der Stroboskopscheibe oder mit einem Digitalzähler auf genau 200 msec eingestellt werden.

Für die Messung der Spurkapazität wird zunächst eine ganze Spur mit 4 µsec Flußwechselabstand (Flußwechselmuster 255 = \$FF) beschrieben. Unmittelbar danach werden drei Disk-Bytes mit 8 µsec Flußwechselabstand als Index aufgezeichnet.

Zum Erzeugen dieser Aufzeichnung wird der Schreibmodus zu 192 (\$C0) gesetzt. Nach dem Schreiben wird gelesen und die Anzahl der Disk-Bytes von Index zu Index gezählt. Die zugehörige Assembler-Routine ist in den Zeilen 122 bis 152 enthalten. Bei der Rückkehr in das BASIC-Programm enthält Adresse 252 und 253 (\$00FC und \$00FD) die Anzahl der Disk-Bytes. Adresse 254 (\$00FE) enthält eine Fehlermarke. Wurde der Index nicht gefunden, weil z.B. keine Diskette eingelegt war oder weil der Schreib- oder Lesekreis defekt war, dann ist die Fehlermarke ungleich null. Das BASIC-Programm gibt dann anstelle der Abweichung das Wort „Fehler“ aus. Voraussetzung für die Messung der Umdrehungsdauer ist also, daß der Schreib-/ Lesekreis einwandfrei funktioniert.

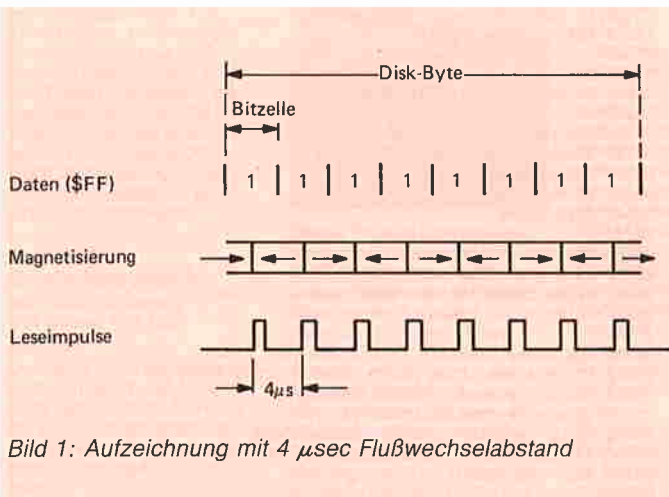


Bild 1: Aufzeichnung mit 4 µsec Flußwechselabstand

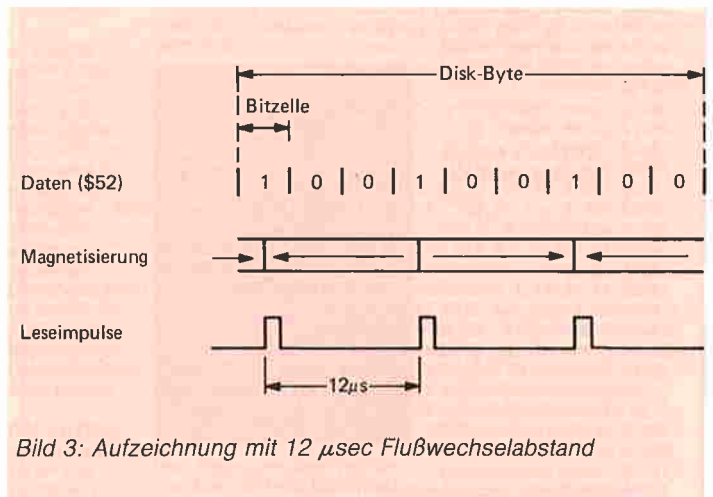


Bild 3: Aufzeichnung mit 12 µsec Flußwechselabstand

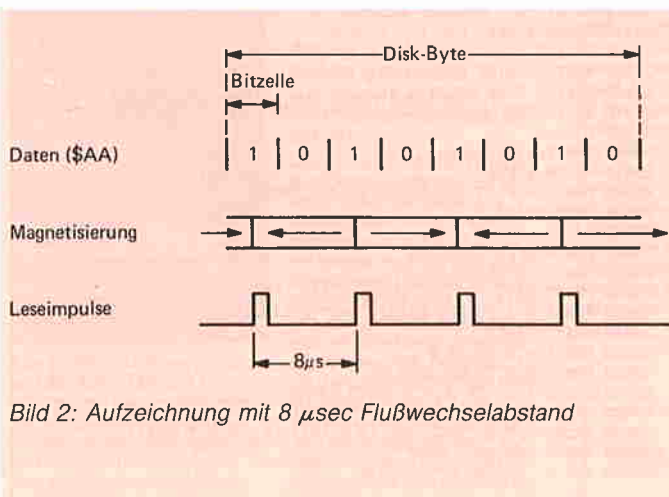


Bild 2: Aufzeichnung mit 8 µsec Flußwechselabstand

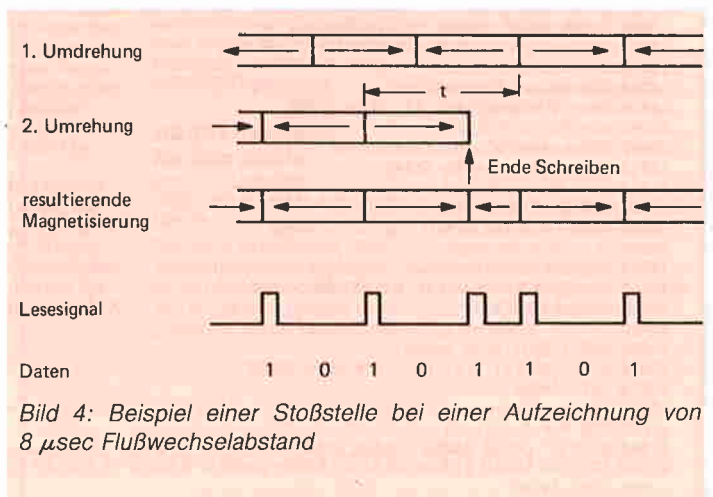


Bild 4: Beispiel einer Stoßstelle bei einer Aufzeichnung von 8 µsec Flußwechselabstand



**Tabelle 1:**

APPLE II DISKETTENSYSTEM TEST

C - Controller-Steckplatz (Slot): 6  
 D - Laufwerk (Drive): 1  
 E - Laufwerk ein-/ausschalten: aus  
 Ø - Positionieren auf Spur Ø  
 I - 1 Spur nach innen  
 A - 1 Spur nach außen  
 P - Positionieren auf Spur:  
 H - hin und her zwischen Spur

T - Schreibschutz testen  
 F - Flußwechselabstand: 4 microsec  
 M - Schreib/Lese Modus: 1 Spur  
 S - Schreiben  
 L - Lesen  
 W - abwechselnd schreiben und lesen

U - Umdrehungsdauer messen  
 J - Spur-Justage: Trigger generieren  
 B - Ausgang nach BASIC

**DISKTEST**

```
10000 REM Testprogramm für Apple II Diskettensysteme
10010 REM von Dipl.-Ing. Gerhard Berg - 2.3.1985
10020 :
10100 POKE 35,24: IF WS > Ø GOTO 10210
10110 HOME : IF PEEK (104) < 64 GOTO 29100
10120 HM = 36864: HMEM: HM
10130 :
10140 AS = 35: REM Anzahl Spuren
10150 PH = 2: REM Phasen pro Spur
10160 BZ = 25: REM Beruhigungszeit in msec
10170 SN = 6: REM Steckplatz-Nr.
10180 LN = 1: REM Laufwerk-Nr.
10190 :
10200 GOSUB 50000:F = Ø: GOSUB 30000
10210 GOSUB 49000: IF F > Ø GOTO 10610
10220 :
10500 REM Hauptschleife
10510 GOSUB 45000: GOSUB 42000:F = Ø
10520 FOR B = 1 TO MB: IF MID$(BT$,B,1) = EZ$ GOTO 10540
10530 NEXT :B = 9
10540 IF FB(B) > Ø AND SN = Ø THEN F = 1: GOTO 10610
10550 IF FB(B) > 1 AND AE = Ø THEN F = 2: GOTO 10610
10560 IF FB(B) > 2 AND S(SN,LN) < Ø THEN F = 3: GOTO 10610
10570 IF FB(B) < 4 GOTO 10600
10580 GOSUB 36000: IF SS = 1 THEN F = 4: GOTO 10610
10590 GOSUB 41000: IF EZ$ = "N" GOTO 10500
10600 ON B GOSUB 11000,12000,13000,14000,15000,16000,17000,
18000,40020,20000,21000,22000,23000,25000,25000,
40020,27000,28000,29000
10610 IF F > Ø THEN GOSUB 40000
10620 GOTO 10500
10630 :
11000 REM Steckplatz auswählen
11010 IF SN > Ø THEN AE = Ø: GOSUB 33000
11020 B1 = 1: GOSUB 44000: GET EZ$:SN = ASC (EZ$) - 48
11030 IF SN < 1 OR SN > 7 THEN PRINT BEL$:: GOTO 11020
11040 GOSUB 42000: PRINT "Bitte warten!";
11050 GOSUB 30000: GOSUB 42000: GOTO 31000
11060 :
12000 REM Laufwerk auswählen
12010 LN = 1 + NOT (LN - 1): GOTO 32000
12020 :
13000 REM Laufwerk ein-/ausschalten
13010 AE = NOT AE: GOTO 33000
13020 :
14000 REM Nach Spur Ø positionieren
14010 S(SN,LN) = MS + 4:S = Ø: GOTO 34000
14020 :
15000 REM 1 Spur nach innen
15010 S = S(SN,LN) + 1: IF S > MS GOTO 40020
15020 GOTO 34000
15030 :
16000 REM 1 Spur nach außen
16010 S = S(SN,LN) - 1: IF S < Ø GOTO 40020
16020 GOTO 34000
16030 :
17000 REM Positionieren
17010 B1 = 7: GOSUB 44000: GOSUB 35000
17020 IF S(SN,LN) < Ø THEN HV = S: GOSUB 14000:S = HV
17030 GOTO 34000
```

```
17040 :
18000 REM Hin und her positionieren
18010 B1 = 8: GOSUB 44000
18020 GOSUB 35000:SH(1) = S: PRINT " und ";
18030 GOSUB 35000:SH(2) = S: GOSUB 43000
18040 IF S(SN,LN) < Ø THEN GOSUB 14000
18050 FOR I = 1 TO 2:S = SH(I): GOSUB 34000
18060 IF PEEK (TD) > 127 GOTO 18080
18070 NEXT I:HV = PEEK (TG): GOTO 18050
18080 B1 = 8: GOSUB 44000: GOTO 42000
18090 :
20000 REM Schreibschutz testen
20010 GOSUB 36000: PRINT "Diskette ist ";
20020 IF SS = Ø THEN PRINT "nicht ";
20030 PRINT "schreibgeschützt.": RETURN
20040 :
21000 REM Flußwechselabstand wählen
21010 FM = FM + 1: IF FM = 4 THEN FM = 1
21020 AB = KS: IF FM = 3 THEN AB = INT (AB * 8 / 9)
21030 B1 = 11: GOSUB 44000
21040 PRINT SPC( FM < 3);FM * 4;" microsec": RETURN
21050 :
22000 REM Schreib-/Lese-Modus wählen
22010 MO = MO + 1: IF MO = 4 THEN MO = 1
22020 B1 = 12: GOSUB 44000: PRINT MO$(MO);: RETURN
22030 :
23000 REM Schreib-Funktion
23010 IF MO < > 2 THEN GOSUB 43000
23020 ON MO GOTO 23100,37000,23200
23030 :
23100 REM Dauernd schreiben
23110 SM = Ø:F1 = FM:A1 = AB: GOSUB 37020: GOTO 42000
23120 :
23200 REM Ganze Diskette beschreiben
23210 FOR S = Ø TO MS: GOSUB 34000: GOSUB 37000
23220 IF PEEK (TD) > 127 GOTO 42000
23230 NEXT : GOTO 42000
23240 :
25000 REM (Schreiben und) lesen
25010 ON MO GOTO 25100,25200,25300
25020 :
25100 REM Dauernd (schreiben und) lesen
25110 GOSUB 43000: POKE 35,22: HOME
25120 GOSUB 25200: PRINT
25130 IF PEEK (TD) < 128 GOTO 25120
25140 GOTO 48000
25150 :
25200 REM Einmal (schreiben und) lesen
25210 IF B = 15 THEN GOSUB 37000
25220 GOSUB 39000: PRINT "Bytes gelesen: ";AB;
25230 PRINT " - Fehler: ";LF:: RETURN
25240 :
25300 REM Ganze Diskette (schreiben und) lesen
25310 GOSUB 43000: POKE 35,22: HOME : GOSUB 47000
25320 FOR S = Ø TO MS: GOSUB 34000
25330 IF B = 15 THEN GOSUB 37000
25340 GOSUB 39000:LF(S) = LF: GOSUB 46000
25350 IF PEEK (TD) > 127 THEN TS = S: GOTO 25400
25360 NEXT :TS = MS
25370 :
25400 GOSUB 42000: VTAB 23: POKE 35,24
25410 PRINT "T - Text-Ausgabe G - Grafik-Ausgabe"
25420 PRINT "D - Drucker-Ausgabe H - Hauptmenü";
25430 GOSUB 45000: FOR I = 1 TO 4
25440 IF EZ$ = MID$( "TDGH",I,1) GOTO 25460
25450 NEXT : PRINT BEL$:: GOTO 25430
25460 TEXT : POKE 35,22 + (I = 4) * 2: HOME
25470 ON I GOTO 25500,25600,25800,49000
25480 :
25500 REM Text-Ausgabe
25510 GOSUB 47000
25520 FOR S = Ø TO TS: GOSUB 36000: NEXT
25530 GOTO 25430
25540 :
25600 REM Drucker-Ausgabe
25610 PR# 1: POKE 35,20
25620 PRINT B$(11);" ";: GOSUB 21040: PRINT
25630 FOR I = 1 TO SP: PRINT "Spur Fehler ";
25640 NEXT : PRINT
25650 FOR I = Ø TO ZE - 1
25660 FOR J = Ø TO MS STEP ZE
25670 S = I + J: IF S > TS GOTO 25700
25680 LF = LF(S):L1 = LEN ( STR$( LF))
25690 PRINT " ";S: SPC( 3 + (S < 10));LF: SPC( 8 - L1);
25700 NEXT : PRINT : NEXT
25710 IN# Ø: PR# Ø: GOTO 25430
```

# BUCH-SHOP

## Apple DOS 3.3

von Ulrich Stiehl  
2. Aufl. 1984, 203 S., kart.,  
DM 28,-

Dies ist die erste deutschsprachige Darstellung des Diskettenbetriebssystems DOS 3.3 für den Apple II/II Plus/IIe, die sich sowohl an Applesoft- als auch an Assembler-Programmierer wendet. Sinngemäß ist das Buch zweigeteilt.

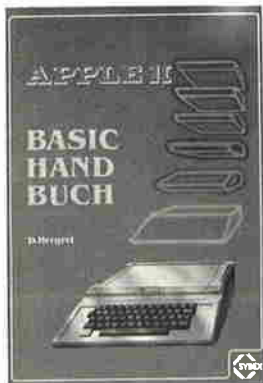
Der erste Teil behandelt ausführlich die dem Applesoft-Programmierer zur Verfügung stehenden DOS-Befehle, wobei die Textfiles wegen ihrer großen Bedeutung und der vergleichsweise komplizierten Handhabung besonders dargestellt werden. Viele Textfile-Tricks werden hier zum ersten Mal geschildert.

Aber auch im zweiten Teil findet der reine Applesoft-Programmierer insbesondere in dem Kapitel „Vermischte Tips, Tricks und Patches“ zahlreiche Anregungen. Im übrigen ist der zweite Teil für Assembler-Programmierer gedacht. Neben einer detaillierten Beschreibung der DOS-Internia enthält dieser Teil elf vollständige RWTS-Anwenderprogramme – z. B. CPM-Refiner, DOS-lose Datendisk, TSL-Maker, File-Reader, Pseudo-Disk-Driver und Fastbrun-Routine –, die Techniken enthüllen, die bislang noch niemals publiziert worden sind. Dieses DOS-Buch ist deshalb der unentbehrliche Begleiter für jeden Apple-Programmierer.

## Apple II Basic Handbuch

von Douglas Hergert  
304 Seiten, 116 Abb.  
DM 32,-

Das Buch ist als Nachschlagewerk konzipiert, daß seinen Platz neben jedem APPLE II, II+ und IIe haben sollte. Es richtet sich an Anfänger und fortgeschrittene Programmierer.



Aus der Praxis heraus präsentiert der Autor Tips und Vorschläge, die das Programmieren leichter und zugleich effizienter machen. Alle Applesoft- und Integer-BASIC-Begriffe sind alphabetisch aufgelistet und werden eingehend erklärt.

Dazu werden alle DOS-Befehle (neben vielen Begriffen der Computerterminologie) vorgestellt.

Beispielprogramme zeigen dem Nutzer, wie jeder Befehl funktioniert und helfen, die richtige Anwendung zu üben. Unter anderem lernt der Leser den besten Weg, um FOR/NEXT-Schleifen und IF/THEN-Entscheidungen für seine Zwecke einzusetzen.

Durch die präzise und leicht verständliche Sprache des Autors werden auch schwierige Befehle einfach in der Anwendung.

## Apple Maschinensprache

von Don und Kurt Inman  
1984, 208 S., zahlr. Abb. und Tabellen, DM 49,-



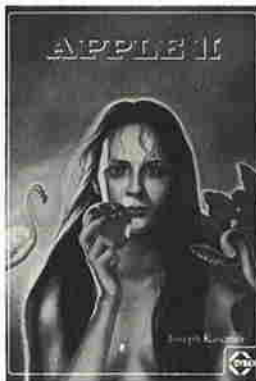
Dieses Buch ist wahrscheinlich die beste Einführung in die 6502-Programmierung für denjenigen Assembler-Anfänger, der zuvor noch nie ein Maschinenprogramm geschrieben hat.

Aus dem Inhalt: Applesoft II BASIC – kurzgefaßt – Alles über Zeichen – Alles über Speicher – Alles über Maschinenbefehle – Maschinenprogramme mit BASIC eingeben – Graphik – Text – Ton – Arithmetik – Was tun mit den Maschinenprogrammen

## Apple II leicht gemacht

von Joseph Kascmer  
1984, 185 S., zahlr. Abb., kart.,  
DM 28,-

Dies ist ein Buch, wie es sich jeder Apple-Anfänger nur wünschen kann: Schrittweise, leichtverständliche Anleitung zum Umgang mit dem Apple mit einigen durchsichtigen, unkomplizierten Beispielen in Applesoft, die ihn nicht Abschrecken, sondern ermutigen sollen, sich mit dem Gerät näher vertraut zu machen. Damit ist „Apple II leicht gemacht“ das ideale Einsteigerbuch für den reinen Anwender, der nicht nur „auf den Knopf drücken“, sondern zumindest einige Details aus der Black Box namens Apple erfahren will.



Aus dem Inhalt: Kontrolle des Geräts – Schreiben und Zeichnen auf dem Bildschirm – Geheimnisvolle Abläufe: Programme – Verschiedene Eingriffsmöglichkeiten – Mobile Speicher: Disketten – Kontrollmöglichkeiten – Das Innenleben

## Apple Assembler

Tips und Tricks  
von Ulrich Stiehl  
1984, 226 S., 3 Abb., kart.,  
DM 34,-

„Apple Assembler“ wendet sich an alle, die bereits Anfängerkenntnisse der 6502-Programmierung haben – z. B. aufgrund des Buches „Apple Maschinensprache“ – und nunmehr ein Nachschlagewerk für ihren Apple II Plus/IIe/IIc suchen, in dem alle wichtigen ROM-Routinen sowie eine Vielzahl sonstiger Hilfsprogramme in einer systematischen Form zusammengestellt werden. Insgesamt umfaßt dieses Buch über 40 Utilities, darunter mehrere völlig neuartige Programme wie Double-Lores, Double Hires, Screen-Format u. a.

Der erste Teil enthält ein Repetitorium der wichtigsten Befehle, Adressierungsarten und sonstigen Besonderheiten des 6502.

Im zweiten Teil werden alle Adressen des Monitors zusammengestellt, die für Assembler-Programmierer von Nutzen sein können. Darüber hinaus findet der Leser Unterroutinen für hexadezimale Addition/Subtraktion/Multiplikation/Division, Binär-Hex-ASCII-Umwandlung usw. Der dritte Teil befaßt sich mit der Speicherverwaltung der Language Card und der IIe-64K-Karte und enthält Move-Programme zum Verschieben von Daten in die und aus der Language Card sowie der 64K-Karte.

Der vierte Teil ist dem Applesoft-ROM gewidmet und listet eine große Anzahl nützlicher Interpreter-Adressen. Bei den Utility-Programmen liegt das Schwergewicht auf Fließkommamathematik einschließlich Print Using.

Der letzte Teil behandelt den Text- und Graphikspeicher. Neben einem professionellen Maskengeneratorprogramm werden auch Routinen zur Double-Lores- und Double-Hires-Grafik vorgestellt.

## Arbeiten mit dem Macintosh

von N. Hesselmann  
416 Seiten, 320 Abb. DM 54,-

Das Buch erklärt den Umgang mit dem Macintosh von Grund auf, wobei auch auf elementare Dinge eingegangen wird, wie z. B. die Benutzung der Tastatur und der Maus, das Einlegen von Disketten und den Systemstart. Ganz besonderes Augenmerk wird auf die Erklärung der speziellen Software-Umgebung des Macintosh gelegt, wobei das Menü- und Fensterkonzept sowie das Anwählen durch Piktogramme gekennzeichnete Funktionen klar dargestellt wird.



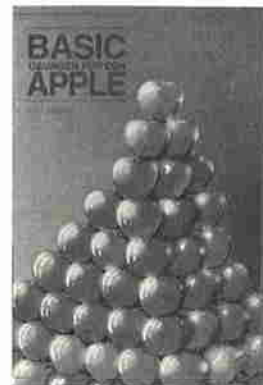
Der Umgang mit den Programmen MacPaint und MacWrite wird erläutert; dies geschieht teilweise anhand von Beispielen, die leicht nachvollzogen werden können. Ein umfangreiches Kapitel ist dem für den Macintosh erhältlichen Microsoft-BASIC gewidmet.

## BASIC Übungen für den Apple

von J. P. Lamoitier  
1983, 252 S., zahlr. Abb., kart.,  
DM 38,-

Das Buch ist konzipiert, allen Apple-Anwendern Applesoft-BASIC durch praktische Übungen an Hand von realen Programmen beizubringen. Daten-

verarbeitung, Statistik, kommerzielle Programme, Spiele und vieles mehr. Jede Übung beinhaltet eine Beschreibung der Problemstellung, eine Analyse der Lösungsmöglichkeiten, ein Flußdiagramm und ein fertiges Programm samt Probelauf.



Aus dem Inhalt: Ihr erstes BASIC-Programm – Flußdiagramme – Übungen mit Integerzahlen – Elementare Beispiele aus der Geometrie – Allgemeine Übungen aus der Datenverarbeitung – Mathematische Berechnungen – Kaufmännische Berechnungen – Spiele – Operations Research – Statistik

## Apple ProDOS für Aufsteiger

Band 1  
von Ulrich Stiehl  
1984, 202 S., kart., DM 28,-

ProDOS ist das neue „professionelle DOS“ (Professional Disk Operating System) für den Apple IIe sowie den mit einer Language Card ausgestatteten Apple II Plus. Band 1 befaßt sich mit den theoretischen Grundlagen von ProDOS, der internen und externen Speicherorganisation und enthält grundlegende Beispielprogramme für Assembler-Programmierer sowie generelle Untersuchungen zum BASIC-SYSTEM. Da ProDOS über erheblich vielfältigere und leistungsfähigere, zugleich jedoch erheblich kompliziertere Dateistrukturen verfügt, sind theoretische Kenntnisse von ProDOS unabdingbar, wenn man die Features von ProDOS voll ausschöpfen will.

Aus dem Inhalt: Ein erster Überblick – ProDOS und DOS 3.3 – Interne Speicherorganisation – Externe Speicherorganisation – MLI (Machine Language Interface) – ProDOS für Applesoft-Programmierer

**Beachten Sie die Buch-Shop-Karte**

```

25720 :
25800 REM Grafik-Ausgabe
25810 VTAB 21:H1 = INT (MS / 40) + 1:H2 = 5 * H1
25820 FOR I = 0 TO TS STEP H2
25830 HTAB 2 + I * 6 / (7 * H1): PRINT I: NEXT
25840 HGR : HCOLOR= 3:SF = 152 / LOG (AB)
25850 FOR I = 0 TO 3:Y = 152 - LOG (10 ↑ I) * SF
25860 HPLLOT 0,Y TO 2,Y: NEXT
25870 FOR I = 0 TO TS STEP H1
25880 X = 10 + I * 6 / H1: HPLLOT X,153 TO X,154
25890 IF I = INT (I / H2) * H2 THEN HPLLOT TO X,157
25900 NEXT
25910 X1 = 6 + H1:X = X1: HPLLOT X1,152
25920 FOR S = 0 TO TS
25930 LF = LF(S): IF LF = 0 THEN LF = 1
25940 Y = 152 - LOG (LF) * SF: HPLLOT TO X,Y
25950 X = X + 6 / H1: HPLLOT TO X,Y: NEXT
25960 HPLLOT TO X,152: HPLLOT TO X1,152: GOTO 25430
25970 :
27000 REM Umdrehungsdauer messen
27010 GOSUB 43000: POKE 35,22: HOME
27020 HGR : HCOLOR= 3:Y = 0
27030 X1 = 2 * KS / 100: REM 1%
27040 FOR X = 140 - X1 TO 140 + X1 STEP X1
27050 HPLLOT X,0 TO X,154: NEXT
27060 VTAB 21: HTAB 2: PRINT "-1%"; TAB( 38);"+1%";
27070 IF PEEK (TD) > 127 GOTO 48000
27080 SM = 192:A1 = KS * 1.1:F1 = 1: GOSUB 37020
27090 CALL UR: VTAB 22: HTAB 18: CALL ZL
27100 IF PEEK (254) = 0 GOTO 27120
27110 PRINT "Fehler": GOTO 27070
27120 BY = PEEK (252) + 256 * PEEK (253) + 1
27130 UT = INT ((BY - KS) * 1000 / KS + 0.5) / 10
27140 UA = ABS (UT):UT$ = STR$ (UA): IF UA > 20 GOTO 27110
27150 IF UA > 0 AND UA < 1 THEN UT$ = "0" + UT$
27160 UT$ = VZ$( SGN (UT) + 2) + UT$
27170 IF UT = INT (UT) THEN UT$ = UT$ + ".0"
27180 PRINT UT$;"%";
27190 X = 140 + (BY - KS) * 2: IF X < 0 THEN X = 0
27200 IF X > 279 THEN X = 279
27210 HPLLOT X,Y:Y = Y + 1: IF Y < 155 GOTO 27070
27220 GOTO 27020
27230 :
28000 REM Trigger für Spur-Justage generieren
28010 IF S(SN,LN) <> JS THEN F = 7: RETURN
28020 GOSUB 43000: POKE 255,190: CALL JR: GOTO 42000
28030 :
29000 REM Ausgang nach BASIC
29010 HOME : POP : END
29020 :
29100 PRINT "Bitte geben Sie ein:": PRINT
29110 PRINT "POKE 104,64": PRINT "POKE 16384,0"
29120 PRINT : PRINT "und laden Sie das Programm neu!"
29130 GOTO 29210
29140 :
29200 HOME : PRINT "Fehler in 'DATA'- Befehlen!"
29210 PRINT CHR$( 7);: CLEAR : END
29220 :
29230 REM -----
29240 :
30000 REM Steckplatz (Slot) auswählen
30010 F$(5) = LEFT$( F$(5),35) + STR$( SN) + "!"
30020 PA = 49152 + 256 * SN: FOR I = 1 TO 7 STEP 2
30030 IF PEEK (PA + I) = CE(I) GOTO 30050
30040 SN = 0:F = 5: RETURN
30050 NEXT :SA = 49280 + 16 * SN
30060 PS = 0: FOR I = 0 TO 255
30070 PS = PS + PEEK (PA + I): NEXT
30080 IF PS <> 31558 THEN F = 6
30090 POKE 250,SN * 16
30100 RETURN
30110 :
31000 REM Steckplatz Nr. ausgeben
31010 IF SN = 0 THEN RETURN
31020 B1 = 1: GOSUB 44000: PRINT SN
31030 :
32000 REM Laufwerk auswählen
32010 B1 = 2: GOSUB 44000: PRINT LN: GOSUB 34050
32020 :
33000 REM Motor ein-/ausschalten
33010 B1 = 3: GOSUB 44000: PRINT AE$(AE)
33020 HV = PEEK (SA + 8 + AE): REM Motor ein/aus
33030 HV = PEEK (SA + 9 + LN): REM Laufwerk 1 / 2
33040 REM In obigen Befehlen keine "POKEs" verwenden!
33050 RETURN
33060 :
34000 REM Positionieren nach Spur 'S'

```

```

34010 IF S = S(SN,LN) THEN RETURN
34020 POKE 251,S * PH: POKE 252,S(SN,LN) * PH
34030 POKE 253,BZ: CALL PR
34040 S(SN,LN) = S: IF B > 13 THEN RETURN
34050 REM Spur # ausgeben
34060 B1 = 7: GOSUB 44000: IF S(SN,LN) < 0 THEN RETURN
34070 PRINT S(SN,LN);: RETURN
34080 :
35000 REM Spur-Nr. eingeben
35010 H = POS (0) + 1: GOTO 35030
35020 PRINT BEL$:
35030 VTAB V: HTAB H: CALL ZL
35040 INPUT "":EZ$: IF EZ$ = "" GOTO 35020
35050 S = VAL (EZ$): IF S < 0 OR S > MS GOTO 35020
35060 IF INT (S) <> S OR STR$( S) <> EZ$ GOTO 35020
35070 VTAB V: HTAB H: CALL ZL: PRINT S: RETURN
35080 :
36000 REM Schreibschutz prüfen
36010 HV = PEEK (SA + 13):SS = PEEK (SA + 14) > 127
36020 HV = PEEK (SA + 12): RETURN
36030 :
37000 REM Schreiben
37010 A1 = AB * 1.1:F1 = FM:SM = 64
37020 GOSUB 38000: POKE 254,SM
37030 CALL SR: RETURN
37040 :
38000 REM Byte-Anzahl und Flußwechselabstand setzen
38010 A1 = 65536 - INT (A1): REM 2'er Komplement
38020 A2 = INT (A1 / 256):A1 = A1 - A2 * 256
38030 POKE 251,FM(F1): POKE 252,A1: POKE 253,A2
38040 RETURN
38050 :
39000 REM Lesen
39010 A1 = AB:F1 = FM: GOSUB 38000: CALL LR
39020 LF = PEEK (254) + 256 * PEEK (255): RETURN
39030 :
40000 REM Fehler-Nachricht ausgeben
40010 GOSUB 42000: FLASH = PRINT F$(F):
40020 NORMAL : PRINT BEL$;: RETURN
40030 :
41000 REM Lösch-Warnung ausgeben
41010 T$ = "Spur " + STR$( S(SN,LN)) + " wird"
41020 IF M0 = 3 AND B < 17 THEN T$ = "Alle Spuren werden"
41030 POKE 35,24: VTAB 23: PRINT T$;" gelöscht!"
41040 PRINT "weiter? (J/N) ";
41050 PRINT BEL$;: GET EZ$: GOSUB 45500
41060 IF EZ$ <> "J" AND EZ$ <> "N" GOTO 41050
41070 :
42000 REM Zeile 23 und 24 löschen
42010 POKE TT,0: VTAB 23: HTAB 1: CALL ZL
42020 VTAB 24: CALL ZL: RETURN
42030 :
43000 REM Stop-Hinweis ausgeben
43010 GOSUB 42000: INVERSE
43020 PRINT "STOP MIT JEDER TASTE";
43030 NORMAL : RETURN
43040 :
44000 REM Cursor zum Ende der Befehlszeile
44010 V = B1 + 2:H = LEN (B$(B1)) + 6
44020 VTAB V: HTAB H: CALL ZL: RETURN
44030 :
45000 REM Zeichen-Eingabe ohne Cursor
45010 EZ = PEEK (TD): IF EZ < 128 GOTO 45010
45020 POKE TT,0:EZ$ = CHR$( EZ - 128)
45030 :
45500 REM Klein- in Großbuchstaben umwandeln
45510 IF EZ$ <= CHR$( 96) THEN RETURN
45520 EZ$ = CHR$( ASC (EZ$) - 32): RETURN
45530 :
46000 REM Fehler pro Spur ausgeben
46010 H1 = INT (S / ZE):V = S - ZE * H1 + 2
46020 H = H1 * 40 / SP + 1: VTAB V: HTAB H + (SP < 4)
46030 PRINT S; TAB( H + 3 + 3 * (SP < 4));LF(S);
46040 RETURN
46050 :
47000 REM Text-Überschrift
47010 SP = INT (MS / 20) + 1
47020 ZE = INT (MS / SP) + 1: INVERSE
47030 FOR I = 0 TO SP - 1: HTAB I * 40 / SP + 1
47040 IF SP < 4 THEN PRINT "SPUR FEHLER";
47050 IF SP >= 4 THEN PRINT "SP.FEH.";
47060 NEXT : NORMAL : RETURN
47070 :
48000 REM Zurück zum Hauptmenü
48010 GOSUB 42000: INVERSE
48020 PRINT "WEITER MIT JEDER TASTE";
48030 NORMAL : GOSUB 45000: TEXT

```



```

48040 :
49000 REM Menü ausgeben
49010 HOME : HTAB 5: INVERSE : PRINT TI$
49020 NORMAL : PRINT
49030 FOR B = 1 TO MB: B$ = MID$( BT$, B, 1)
49040 IF B$ < > " " THEN PRINT B$, " - ", B$(B);
49050 PRINT : NEXT
49060 GOSUB 21020: GOSUB 22020: GOTO 31000
49070 :
49080 REM -----
49090 :
50000 REM Programm-Initialisierung
50010 TI$ = "APPLE II DISKETTENSYSTEM TEST"
50020 SPEED= 100: HOME
50030 HTAB 5: INVERSE : PRINT TI$: NORMAL : PRINT
50040 PRINT TAB( 6); "von Gerhard Berg - 2.3.1985"
50050 VTAB 23: HTAB 1: INVERSE
50060 PRINT "PROGRAMM-DISKETTE AUS LAUFWERK NEHMEN"
50070 HTAB 5: PRINT "UND LEERE DISKETTE EINLEGEN!";
50080 NORMAL : SPEED= 255
50090 VTAB 12: HTAB 13: PRINT "Bitte warten"
50100 :
51000 AE = 0: FM = 1: MO = 2: KS = 6378: JS = 16 + 16 * ( AS >
40)
51010 MS = AS - 1: DIM LF(MS), S(7, 2)
51020 FOR I = 0 TO 7: S(I, 1) = - 1: S(I, 2) = - 1: NEXT
51030 BEL$ = CHR$( 7): ZL = 64668
51040 FM(1) = 255: FM(2) = 170: FM(3) = 146
51050 CE(1) = 32: CE(3) = 0: CE(5) = 3: CE(7) = 60
51060 TD = 49152: TT = 49168: TG = 49216
51070 LR = HM: SR = HM + 33: UR = HM + 110
51080 JR = HM + 159: PR = HM + 186
51090 AES(0) = "aus": AES(1) = "ein"
51100 VZ$(1) = "-": VZ$(2) = " ": VZ$(3) = "+"
51110 BT$ = "CDEIAPH TPMSLW UJB"
51120 MB = LEN (BT$): DIM B$(MB), FB(MB)
51130 :
52000 FOR I = 1 TO MB: READ B$(I): NEXT
52010 DATA "Controller-Steckplatz (Slot) #:"
52020 DATA "Laufwerk (Drive) #:"
52030 DATA "Laufwerk ein-/ausschalten:"
52040 DATA Positionieren auf Spur 0
52050 DATA 1 Spur nach innen, 1 Spur nach außen
52060 DATA "Positionieren auf Spur:"
52070 DATA hin und her zwischen Spur
52080 DATA " ", Schreibe-/Schutz testen
52090 DATA "Flußwechselabstand:"
52100 DATA "Schreib/Lese Modus:"
52110 DATA Schreiben, Lesen
52120 DATA abwechselnd Schreiben und Lesen
52130 DATA " ", Umdrehungsdauer messen
52140 DATA "Spur-Justage: Trigger generieren"
52150 DATA Ausgang nach BASIC
52160 :
52500 FOR I = 1 TO MB: READ FB(I): NEXT
52510 DATA 0, 1, 1, 2, 3, 3, 2, 2, 0, 2, 0, 0, 4, 3, 4, 0, 4, 3, 0
52520 :
53000 FOR F = 1 TO 7: READ F$(F): NEXT
53010 DATA KEIN CONTROLLER AUSGEWÄHLT!
53020 DATA LAUFWERK IST NICHT EINGESCHALTET!
53030 DATA KOPFPOSITION IST UNDEFINIERT!
53040 DATA DISKETTE IST SCHREIBGESCHÜTZT!
53050 DATA "KEIN DISK-CONTROLLER IN STECKPLATZ "
53060 DATA FEHLER IN PRÜFSUMME DES BOOT-PROM'S!
53070 DATA SPUR-JUSTAGE AUF SPUR
53080 F$(7) = F$(7) + " " + STR$( JS) + "!"
53090 :
54000 FOR I = 1 TO 3: READ MO$(I): NEXT
54010 DATA dauernd, 1 Spur, ganze Diskette
54020 :
55000 REM Assemblerprogramm abspeichern
55010 PS = 0: FOR I = 36864 TO 37174
55020 READ HV: POKE I, HV: PS = PS + HV: NEXT
55030 DATA 166, 250, 169, 0, 133, 254, 133, 255
55040 DATA 164, 252, 189, 140, 192, 16, 251, 197
55050 DATA 251, 240, 6, 230, 254, 208, 2, 230
55060 DATA 255, 200, 208, 238, 230, 253, 208, 234
55070 DATA 96, 166, 250, 36, 254, 165, 251, 221
55080 DATA 141, 192, 157, 143, 192, 221, 140, 192
55090 DATA 8, 40, 234, 234, 8, 40, 201, 146
55100 DATA 208, 2, 230, 255, 157, 141, 192, 221
55110 DATA 140, 192, 112, 10, 234, 172, 0, 192
55120 DATA 16, 234, 189, 142, 192, 96, 230, 252
55130 DATA 208, 226, 230, 253, 208, 224, 44, 254
55140 DATA 0, 16, 239, 160, 3, 169, 170, 157
55150 DATA 141, 192, 221, 140, 192, 72, 104, 72
55160 DATA 104, 136, 240, 222, 208, 239, 166, 250

```

```

55170 DATA 169, 3, 133, 254, 198, 254, 240, 36
55180 DATA 160, 0, 132, 253, 189, 140, 192, 16
55190 DATA 251, 200, 208, 4, 230, 253, 48, 20
55200 DATA 201, 255, 208, 240, 189, 140, 192, 16
55210 DATA 251, 201, 170, 240, 223, 200, 208, 244
55220 DATA 230, 253, 16, 240, 132, 252, 96, 166
55230 DATA 250, 189, 140, 192, 16, 251, 44, 0
55240 DATA 192, 48, 243, 201, 255, 208, 242, 44
55250 DATA 48, 192, 44, 64, 192, 32, 8, 145
55260 DATA 240, 229, 169, 0, 133, 254, 165, 252
55270 DATA 133, 255, 56, 229, 251, 240, 49, 176
55280 DATA 6, 73, 255, 230, 252, 144, 4, 105
55290 DATA 254, 198, 252, 197, 254, 144, 2, 165
55300 DATA 254, 201, 12, 176, 1, 168, 56, 32
55310 DATA 252, 144, 185, 31, 145, 32, 19, 145
55320 DATA 165, 255, 24, 32, 254, 144, 185, 43
55330 DATA 145, 32, 19, 145, 230, 254, 208, 198
55340 DATA 32, 8, 145, 24, 165, 252, 41, 3
55350 DATA 42, 5, 250, 170, 189, 128, 192, 96
55360 DATA 164, 253, 169, 10, 32, 19, 145, 136
55370 DATA 208, 248, 96, 162, 18, 202, 208, 253
55380 DATA 234, 56, 233, 1, 208, 245, 96, 1
55390 DATA 48, 40, 36, 32, 30, 29, 28, 28
55400 DATA 28, 28, 28, 112, 44, 38, 34, 31
55410 DATA 30, 29, 28, 28, 28, 28, 28, 28
55500 IF PS < > 45695 GOTO 29200
55510 IN# 0: PR# 0
55520 POKE 35, 22: WS = 1: RETURN

```

## DISKTEST.START

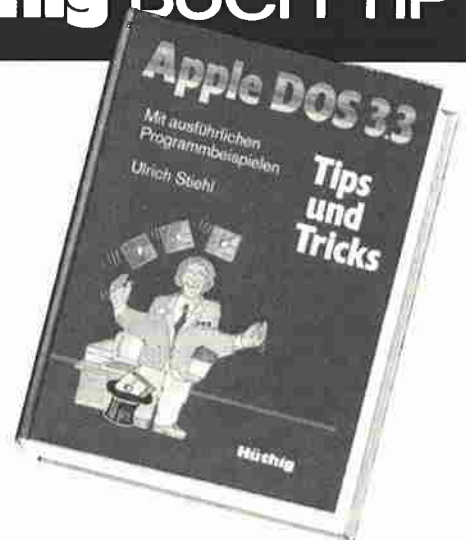
```

POKE 104, 64
POKE 16384, 0
RUN DISKTEST

```



## Hüthig BUCH-TIP



### Apple DOS 3.3 — Tips und Tricks

von U. Stiehl

2. Aufl. 1984, 216 S., mit zahlreichen,  
ausführlich kommentierten Programm-  
listings, kart., DM 28,—  
ISBN 3-7785-1049-5

Dr. Alfred Hüthig Verlag · Postf. 10 28 69 · 6900 Heidelberg 1

# ProDOS für Anfänger

## Teil 3: Geheimnisse von BSAVE und BLOAD

von Ulrich Stiehl

### 1. FID in Applesoft?

Wenn Sie bislang noch nicht vom BASIC.SYSTEM überzeugt waren, dann sind Sie es sicherlich, wenn Sie den nachfolgenden Aufsatz gelesen haben, in dem ich Ihnen die bislang noch nicht publizierten Eigenschaften der zwei leistungsfähigsten BASIC.SYSTEM-Befehle vorstellen werde.

Sie kennen alle das alte FID (File Developer) für DOS 3.3, das zum Kopieren von DOS-Dateien dient. Dieses Maschinenprogramm hat eine Länge von ca. 4800 Bytes. Nun stellen Sie sich einmal vor, Sie sollten dieses Programm (fast) ausschließlich in Applesoft-Basic (mit höchstens ganz kurzen Maschinenroutinen) schreiben. Wäre dies möglich? Wie groß wäre dann das Applesoft-Programm? Und wie würden Sie vorgehen, damit T-, B- und A-Dateien beliebiger Länge kopiert werden können? Sie werden jetzt mit Recht vermuten, daß nicht nur ein sehr großes, sondern auch ein sehr langsames Applesoft-Programm entstehen würde.

Nicht so unter dem BASIC.SYSTEM von ProDOS! Hier ist es tatsächlich möglich, ein Kopierprogramm für Dateien beliebiger Art und Größe fast ausschließlich in Applesoft zu schreiben, das nicht nur sehr kompakt (weniger als 1800 Bytes), sondern auch relativ schnell ist. Dies ist allein aufgrund des sehr leistungsfähigen BSAVE/BLOAD-Befehls möglich.

### 2. BSAVE und BLOAD

#### 2.1. Speicherbereich

Der Speicher des Apple IIe/IIc/II Plus läßt sich abstrakt als ein Kontinuum von Speicherstellen definieren, die von \$0000-\$FFFF bzw. von 0-65535 durchnummeriert sind. Dabei bezeichnen wir als Speicherbereich einen durchgehenden Abschnitt des Gesamtspeichers, z.B. den Speicherbereich \$2000-\$3FFF (HGR Seite 1). Die Anfangsadresse dieses HGR1-Bereiches ist \$2000, die Endadresse \$3FFF und die Länge \$2000. Bei der Längenberechnung beachte man immer, daß das erste Byte mitgezählt werden muß. Ein einfaches Beispiel:

Speicherbereich: 768-770

Anfangsadresse: 768

Endadresse: 770

Länge: 3 (768 + 769 + 770)

Die Länge errechnet sich demnach durch die Formel

Endadresse - Anfangsadresse + 1,

d.h. hier konkret

$770 - 768 + 1 = 3$

Vertiefend können wir festhalten: Ein Speicherbereich ist eine kontinuierliche Folge von Speicherstellen und läßt sich durch Anfangsadresse, Endadresse und Länge charakterisieren. Im Grenzfall ist die Länge = 1; dann fallen Anfangsadresse und Endadresse zusammen. Jede Speicherstelle enthält einen Wert (= Byte) im Bereich 0-255 bzw. \$00-\$FF. Die einzel-

nen Speicherstellen haben absolute Speicherplatznummern, weil der Gesamtspeicher von 0-65535 durchnummeriert ist. Daneben gibt es noch relative Speicherplatznummern (= Byte-Offset oder kurz Offset) im Bereich 0 bis (Länge - 1). Nehmen wir hierzu an, daß die absoluten Speicherstellen 768-770 die Bytes \$C1, \$C2 und \$C3 (entspricht ASCII „A“, „B“ und „C“) enthalten. Dann gilt im einzelnen:

$\$0300 = 768 - \$0000 = 0$  (\$C1=„A“)

$\$0301 = 779 - \$0001 = 1$  (\$C2=„B“)

$\$0302 = 770 - \$0002 = 2$  (\$C3=„C“)

Die absolute Speicherstelle 768 ist zugleich die nullte Speicherstelle, wenn die Zählung mit 0 bei 768 beginnt. Die nullte – oder bei „normaler“ Zählweise „erste“ – Speicherstelle (mit dem Offset = Abstand = 0) enthält „A“, die 1. Speicherstelle (Offset = 1) enthält „B“ und die 2. und letzte Speicherstelle (Offset = 2) enthält „C“. Der Byte-Offset liegt damit stets im Bereich 0 bis Länge minus 1.

#### 2.2. Datei

Ziehen wir nunmehr zu den weiteren Erörterungen das **Bild 1** heran. In der oberen Bildhälfte wird ein Teil des RAM-Speichers, nämlich der für BSAVE und BLOAD hauptsächlich in Frage kommende Speicherteil \$0800-\$95FF dargestellt. Als Speicherauszug (= auf Diskette zu speichernder Bereich) wählen wir \$2000-\$3FFF (HGR1).

Während der RAM-Bereich als eine kontinuierliche Folge von Bytes im *internen* Speicher definiert werden kann, läßt sich die Diskettendatei bzw. der *File* als eine kontinuierliche Folge von Bytes im *externen* Speicher (= Datenträger: Diskette, Festplatte, RAM-Disk usw.) charakterisieren. Daß eine Datei letztlich in 512-Byte-Blocks unterteilt ist, die physisch auf der Diskette verstreut sein können und insofern kein Byte-Kontinuum darstellen, ist zunächst irrelevant.

Die Bytes einer Datei sind absolut nummeriert vom nullten bis zum letzten Byte, denn beim BSAVE/BLOAD-Befehl bezieht sich der Byte-Offset immer auf den Anfang der Datei. Betrachten wir hierzu die untere Hälfte von Bild 1. Die dort dargestellte Datei namens FILE hat eine Länge L von \$6000 Bytes, die von Byte-Offset B = \$0000 = absolut nulltes Byte = Dateianfang bis Byte-Offset B = \$5FFF = absolut letztes Byte = Dateiende durchnummeriert sind. Ähnlich wie der Gesamtspeicher in Speicherbereiche unterteilt werden kann, läßt sich auch eine Gesamtdatei in Dateiabschnitte aufteilen. Im Bild 1 ist der Bereich Byte-Offset \$4000 bis \$5FFF ein solcher Teilabschnitt.

Der sog. **EOF** (= End of File = ENDFILE im ProDOS-Directory) hat eine doppelte Bedeutung: Zum einen versteht man unter EOF (als Kardinalzahl) die Gesamtzahl aller Bytes einer Datei, also die Dateilänge. Zum anderen bedeutet EOF (als Ordinalzahl) die Nummer desjenigen Bytes, das dem letzten Byte der Datei folgt bzw. folgen würde. Man beachte also, daß der Begriff EOF nicht mit dem Begriff Dateiende identisch ist.

Nun kommen wir endlich zum BSAVE/BLOAD-Befehl. Der **BSAVE**-Befehl überträgt einen (beliebigen) Speicherbereich als Speicherauszug in einen (beliebigen) Teilbereich einer Datei, während der **BLOAD**-Befehl einen (beliebigen) Teilbereich einer Datei in einen (beliebigen) Speicherbereich einlädt. Im einzelnen gibt es folgende Parameter:

- A = Anfangsadresse des Speicherbereichs
- E = Endadresse des Speicherbereichs
- L = Länge des Speicherbereichs
- B = absoluter Byte-Offset der Datei
- T = Dateityp

**A- und E-Parameter:** Beim BSAVE kann theoretisch jede Anfangs- und Endadresse im Bereich \$0000-\$BFFF angegeben

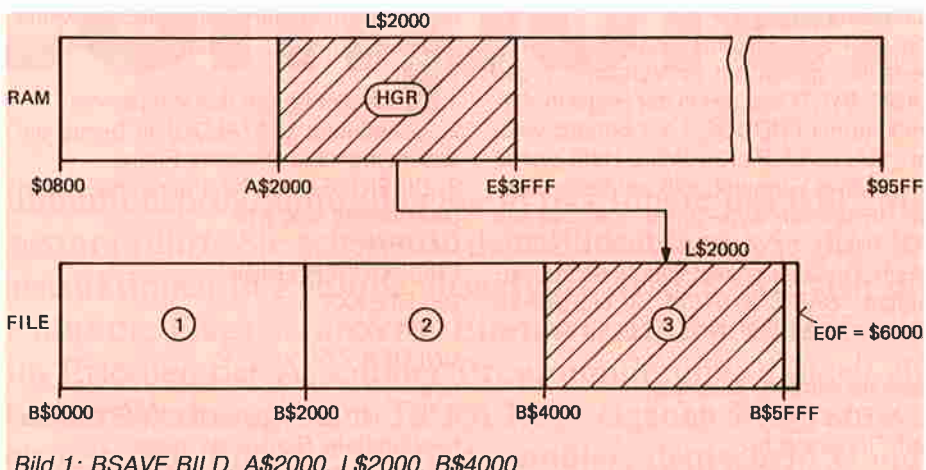


Bild 1: BSAVE BILD, A\$2000, L\$2000, B\$4000

werden. Wegen BLOAD sind praktisch jedoch nur die Bereiche \$0300-\$03CF sowie \$0800-\$95FF zulässig, weil \$0000-\$02FF sowie \$9600-\$BFFF vom BASIC-.SYSTEM als belegt markiert werden.

**L-Parameter:** Die Länge ergibt sich aus  $(E - A + 1)$ . Die maximale Länge beträgt theoretisch \$FFFF = 65535. Beim BSAVE muß man neben A *entweder* E *oder* L angeben, also niemals beide gleichzeitig! Wenn man beim BLOAD A, E und L wegläßt, so gelten als Ersatz die beim letzten BSAVE benutzten A/E/L-Werte. Wurde eine Datei *gerade* mit BLOAD geladen, so kann sie *unmittelbar* danach ohne A, E oder L mit BSAVE zurückgespeichert werden (vgl. Bug weiter unten).

**B-Parameter:** Da unter ProDOS eine einzelne Datei 16M groß sein kann, kann der B-Parameter Werte im Bereich \$000000 bis \$FFFFFF einnehmen. Wenn B fehlt, wird automatisch B = 0 angenommen. Mit B meint man stets den Byte-Offset in der externen Datei und niemals den o.g. Byte-Offset im internen Speicherbereich. Die Werte für A, E, L und B können wahlweise als Dezimal- oder Hexadezimalzahlen (mit vorangestelltem \$-Zeichen) eingegeben werden.

**T-Parameter:** Dieser Parameter für die diversen Dateitypen (TBIN, TTXT usw.) ist nur dann erforderlich, wenn eine Nicht-BIN-Datei bearbeitet werden soll. Wenn nämlich der T-Parameter fehlt, wird automatisch eine BIN-Datei (Binärdatei, B-Datei) angenommen (daher auch die Befehlswörter B-LOAD und B-SAVE). Nicht-BIN-Dateien müssen jedoch erst mit dem CREATE-Befehl erzeugt werden, bevor sie mit BSAVE/BLOAD bearbeitet werden können.

### Beispiele

**BSAVE BILD, A\$2000, L\$2000** speichert die BIN-Datei BILD (ab Byte-Offset 0, weil der B-Parameter fehlt). Würde eine BIN-Datei namens BILD noch nicht auf der Diskette existieren, so würde sie vom BASIC.SYSTEM zunächst automatisch angelegt (= „kreiert“).

**BLOAD BILD** lädt mangels weiterer Parameter die *gesamte* BILD-Datei wieder in den *alten* Speicherbereich \$2000-\$3FFF.

**BLOAD BILD, A\$4000** lädt die BILD-Datei nunmehr in den HGR2-Bereich ab \$4000.

**CREATE TEXT, TTXT**  
**BSAVE TEXT, TTXT, A768, E770** legt zunächst eine TXT-Datei namens TEXT an und speichert dann darauf, wenn wir das obige Beispiel aufgreifen, die Buchstaben „ABC“ ab Byte-Offset 0 ab.

**BSAVE TEXT, TTXT, B3, A768, L3** hängt „ABC“ an das Ende der TXT-Datei an, die nunmehr den Inhalt „ABCABC“ hat.

**BLOAD TEXT, TTXT, A780, B3** lädt die drei letzten Buchstaben „ABC“ der TXT-Datei in den Speicherbereich 780-782.

**BLOAD PRODOS, A\$2000, TSYS** lädt die SYS-Datei PRODOS in den Speicher ab \$2000. Da der B-Parameter fehlt, wird mit dem nullten Byte der Datei begonnen. Und da weder der E- noch der L-Parameter spezifiziert wurde, wird die *gesamte* Datei eingelesen.

Der **BILDTEST** (s. Listing), der mit dem **Bild 1** korrespondiert, packt 3 HGR1-Bilder in eine einzige BIN-Datei. Bild 1 veranschaulicht den BSAVE-Vorgang für das dritte HGR1-Bild.



### 2.3. BSAVE-Bug 1.0

Die oben geschilderten BSAVE-BLOAD-Verfahren gelten nur ab Version 1.1 des BASIC.SYSTEMs, das in der Regel in Verbindung mit PRODOS 1.1.1 benutzt wird. Im „CALL A.P.P.L.E.“ (März-Heft) wurde irrigerweise vermutet, daß nur beim zweiten BSAVE ein Byte-Offset über das alte EOF hinaus möglich sei. Dies ist jedoch falsch. Genauer gesagt konnte beim „verbugten“ BASIC.SYSTEM 1.0 ein BSAVE über den alten EOF hinaus nur für den Fall  $B \leq L$

realisiert werden. Beispiele:

BSAVE XXX, A5000, B0, L2  
ENDFILE danach 2

BSAVE XXX, A5000, B1, L2  
ENDFILE danach 3

BSAVE XXX, A5000, B2, L2  
ENDFILE danach 4

BSAVE XXX, A5000, B3, L2  
ENDFILE danach wieder 4,

weil hier  $B > L$  (Bug)!

Ferner unterscheidet sich der neue 1.1-BSAVE vom alten 1.0-BSAVE darin, daß der 1.1-BSAVE erstens im Falle von  $L <$  momentanes EOF

die Altdatei nicht kürzt bzw. schrumpfen läßt und daß zweitens über den momentanen EOF hinaus BSAVES mit

$B > EOF$

möglich sind, wodurch „Loch“-Dateien entstehen können. Beispiele:

BSAVE XXX, A5000, L1, B\$FFFFFF  
erzeugt unter dem BASIC.SYSTEM 1.1 eine BIN-Datei mit einem ENDFILE von 16777215 Bytes (=  $16M - 1$ ). Unter Version 1.0 würde hier eine Fehlermeldung produziert. Für Interessierte: Diese Datei belegt 5 Blocks, und zwar 2 (!) Datenblocks sowie 3 Index-Blocks.

BSAVE YYY, A\$2000, L\$4000

BSAVE YYY, A\$2000, L\$2000

würde unter Version 1.0 nach dem zweiten BSAVE die Dateilänge wieder auf L\$2000 schrumpfen lassen. Unter Version 1.1 würde weiterhin eine Länge von \$4000 ausgewiesen.

### 3. KOPY und BATCHKOPY

Die nachfolgenden zwei Dateikopierprogramme demonstrieren die ungewöhnliche Leistungsfähigkeit des BASIC.SYSTEMs 1.1.

#### 3.1. KOPY

KOPY wird mit RUN KOPY gestartet. Es schaltet zunächst die 80-Zeichenkarte ein – dies kann man streichen – und meldet

sich dann mit einem spartanischen Menü „BEFEHL (K/C/E):“. Man hat nun 4 Optionen:

1. E bedeutet E(nde) des Programms.

2. C bedeutet C(ATALOG) in bezug auf das momentane Directory-Präfix.

3. DIREKTBEFEHL führt einen beliebigen Direktbefehl aus, z.B.

CAT/RAM

PREFIX/USERS.DISK

DELETE XXX

LOCK YYY

UNLOCK ZZZ

usw.

4. K... ist der eigentliche K(OPY)- oder Kopierbefehl. Die Syntax lautet

K/VOLUME1/FILE1,/VOLUME2/FILE2

d.h. „K“ + „vollständiger Pfadname der Ausgangsdatei“ + „/“ + „vollständiger Pfadname der Zieldatei“. Die eigentlichen File-Namen können identisch oder auch verschieden sein, so daß mit dem Kopiervorgang gleichzeitig eine Datei umbenannt werden kann. Nehmen wir an, daß sich die Datei „PRODOS“ auf der Diskette mit dem Volume-Namen „USERS.DISK“ befindet und auf das Subdirectory „SUBDIR“ der RAM-Disk mit dem Volume-Namen „RAM“ kopiert werden soll. Dann lautet der Kopierbefehl

K/USERS.DISK/PRODOS,/RAM/SUBDIR/PRODOS

Man beachte, daß innerhalb des Befehls keine Leertasten zulässig sind und daß die Volume- bzw. Subdirectory-Namen durch Schrägstriche abgegrenzt werden müssen.

Die Anwendung des Programms KOPY wird dann empfohlen, wenn *einzelne* Dateien kopiert werden sollen.

#### 3.2. BATCHKOPY

Die Optionen 1-3 sind bei BATCHKOPY dieselben wie bei KOPY. BATCHKOPY empfiehlt sich anstelle von KOPY dann, wenn eine Vielzahl von Dateien kopiert werden soll. Zu diesem Zweck gibt man beim Kopierbefehl zunächst ein nacktes „K“ ein, worauf man die Ausgangs- und Zielpräfixe festlegen kann, z.B.

VON PREFIX/USERS.DISK

NACH PREFIX/RAM

Die Schrägstriche vor dem ersten und nach dem letzten Directory-Namen werden automatisch ergänzt. Danach wird das gesamte Ausgangsdirectory Name für Name angezeigt, und man braucht nunmehr nur noch durch „J(a)“ oder „N(ein)“ bestimmten, welche der einzelnen Dateien kopiert werden sollen.

BATCHKOPY ist fast genauso schnell bzw. langsam wie der FILER der Firma Apple, der allerdings 15mal so umfang-

reich ist! Wer ein noch schnelleres Kopierprogramm benötigt, der verwende das in „ProDOS für Aufsteiger, Band 2“ abgedruckte Kopierprogramm **PROFID**, das als reines Maschinenprogramm noch kürzer und zugleich ca. 40% schneller ist. In der Kürze liegt die Würze!

Die Programme KOPY und BATCHKOPY kommen nicht ganz ohne Maschinenroutinen aus. Insbesondere der sog. Get- und Set-File-Info-MLI-Befehl mußte in Assembler geschrieben werden. Die entscheidende Stelle in den beiden Applesoft-Programmen ist jeweils die Zeile 72, die den schubweisen BLOAD- und BSAVE-Befehl ausführt. Dies wäre unter DOS 3.3 undenkbar gewesen. Denken Sie jedoch daran, daß KOPY und BATCHKOPY nur unter dem BASIC.SYSTEM 1.1 funktionieren. (Das o.g. PROFID funktioniert unter allen ProDOS-Versionen.)

Wer das Applesoft-Programm näher betrachtet, wird übrigens feststellen, daß ich eine Schwäche für ON-GOTO-Konstruktionen habe, die an die Stelle des fehlenden IF-THEN-ELSE-Befehls treten.

### 4. BSAVE-Bug 1.1.

Vollziehen Sie folgenden Test:

1. POKE 5000, 1: POKE 10000, 2

2. BSAVE XXX, A5000, L1

3. BSAVE XXX, A10000, L1

4. POKE 10000, 0

5. BLOAD XXX

6. PRINT PEEK (5000)

7. PRINT PEEK (10000)

8. CATALOG

Sind Sie überrascht?

Der 1.1-BSAVE-Befehl ändert die beim *ersten* BSAVE registrierte Anfangsadresse später *nie mehr*. Deshalb muß man stets den A-Parameter hinzufügen, sonst können unheilvolle Dinge passieren. Es wurde also bei 1.1 ein alter Bug beseitigt und ein neuer eingebaut. Dieser Bug kommt bei KOPY und BATCH KOPY nicht zum Tragen.





## KOPY

(Warnung: KOPY funktioniert nur mit BASIC.SYSTEM 1.1!)

```

10 PRINT CHR$(4);"PR#3": PRINT : IF PEEK(116) < 128 THEN
  PRINT "KEIN PUFFER!": END
12 DATA 32,0,191,196,55,3,176,38,169,10,141,73,3,32,0,
  191,196,73,3,176,25,162,3,189,58,3,157,76,3,202,16,247,
  169,7,141,73,3,32,0,191,195,73,3,176,1,96,32,218,253,32
14 DATA 58,255,76,208,3,10,0,2,0,0,0,0,0,0,0,0,0,0,0,0,
  0,0,128,2: FOR X = 768 TO 843: READ Y: POKE X,Y: NEXT:
  REM GETINFO
16 DATA 32,6,227,32,190,222,32,227,223,32,108,221,
  133,133,134,32,44,213,200,32,233,227,76,154,218:
  FOR X = 900 TO 925: READ Y: POKE X,Y: NEXT : REM INALL
18 ONERR GOTO 24
20 HOME : INVERSE : PRINT "*** KOPY ***":
  PRINT "* U.STIEHL *": NORMAL : GOTO 26
22 PRINT "SYNTAX?": GOTO 26
24 ON E = 1 AND PEEK(222) = 19 GOTO 62:
  PRINT "FEHLER "; PEEK(222);"!"
26 CLEAR : PRINT : PRINT "BEFEHL (K/C/E)": : CALL 900,B$:
  ON B$ = "" GOTO 26: ON LEFT$(B$,1) = "K" GOTO 34:
  IF B$ = "E" THEN POKE 216,0: END
28 IF B$ = "C" THEN PRINT CHR$(4);"CATALOG": GOTO 26
30 PRINT CHR$(4);B$: GOTO 26
32 REM SYNTAX:K/V1/N1./V2/N2
34 B = LEN(B$): IF B < 10 THEN 22
36 FOR X = B TO 2 STEP -1: IF MID$(B$,X,1) = ","
  THEN N2$ = MID$(B$,X+1,B-X):
  N1$ = MID$(B$,2,X-2): X = 2
38 NEXT : ON N1$ = "" OR N1$ = N2$ GOTO 22:N1 = LEN(N1$):
  N2 = LEN(N2$): ON N1 < 4 OR N2 < 4 GOTO 22:
  ON LEFT$(N1$,1) <> "/" OR LEFT$(N2$,1) <> "/" GOTO 22
40 FOR X = N1 TO 2 STEP -1: IF MID$(N1$,X,1) = "/"
  THEN P1$ = LEFT$(N1$,X):F1 = N1 - X:
  F1$ = RIGHT$(N1$,F1): X = 2
42 NEXT : IF P1$ = "" THEN 22
44 FOR X = N2 TO 2 STEP -1: IF MID$(N2$,X,1) = "/"
  THEN P2$ = LEFT$(N2$,X):X = 2
46 NEXT : IF P2$ = "" THEN 22
48 REM DIR-READ
50 B$ = CHR$(4) + "VERIFY": PRINT B$:P2$: PRINT B$:P1$:
  PRINT B$:N1$: PRINT CHR$(4);"OPEN":P1$;"TDIR":
  PRINT CHR$(4);"READ":P1$: INPUT B$: INPUT B$: INPUT B$
52 INPUT B$: IF MID$(B$,2,F1) = F1$ AND MID$(B$,F1+2,1)
  = " " THEN PRINT CHR$(4);"CLOSE": GOTO 58
54 ON B$ <> "" GOTO 52: PRINT CHR$(4);"CLOSE":
  PRINT "NAME?": GOTO 22
56 REM KOPIEREN
58 LL = VAL(MID$(B$,65,7)):T$ = ",T" + MID$(B$,18,3):
  B = 0:A = 6144:L = 24576: REM $1800+$6000=$7800
60 E = 1: PRINT CHR$(4);"CREATE":N2$:T$: GOTO 64:
  REM E=ERROR
62 E = 0: PRINT CHR$(4);"DELETE":N2$:
  PRINT CHR$(4);"CREATE":N2$:T$
64 IF LL < L THEN L = LL: ON L > 0 GOTO 70: RUN 26
66 GOSUB 72:B = B + L: ON B + L < LL GOTO 66:L = LL - B:
  IF L = 0 THEN RUN 26: REM LOOP
68 REM SETINFO
70 GOSUB 72: POKE 512,N1: FOR X = 1 TO N1:
  POKE 512 + X, ASC(MID$(N1$,X,1)): NEXT:
  POKE 640,N2: FOR X = 1 TO N2: POKE 640 + X,
  ASC(MID$(N2$,X,1)): NEXT : CALL 768: RUN 26
72 PRINT CHR$(4);"BLOAD":N1$:T$;"A":A;"B":B;"L":L:
  PRINT CHR$(4);"BSAVE":N2$:T$;"A":A;"B":B;"L":L:
  RETURN

```

## BATCHKOPY

(Warnung: BATCHKOPY funktioniert nur mit BASIC.SYSTEM 1.1!)

```

10 PRINT CHR$(4);"PR#3": PRINT : IF PEEK(116) < 128 THEN
  PRINT "KEIN PUFFER!": END
12 DATA 32,0,191,196,55,3,176,38,169,10,141,73,3,32,
  0,191,196,73,3,176,25,162,3,189,58,3,157,76,3,202,16,247,
  169,7,141,73,3,32,0,191,195,73,3,176,1,96,32,218,253,32
14 DATA 58,255,76,208,3,10,0,2,0,0,0,0,0,0,0,0,0,0,0,
  0,0,0,0,0,0,0,0,128,2: FOR X = 768 TO 843:
  READ Y: POKE X,Y: NEXT : REM GETINFO
16 DATA 32,6,227,32,190,222,32,227,223,32,108,221,133,
  133,132,134,32,44,213,200,32,233,227,76,154,218:
  FOR X = 900 TO 925: READ Y: POKE X,Y: NEXT : REM INALL
18 ONERR GOTO 24
20 HOME : INVERSE : PRINT "*BATCH-KOPY*":
  PRINT "* U.STIEHL *": NORMAL : GOTO 26

```

```

22 PRINT "SYNTAX?": GOTO 26
24 ON E = 1 AND PEEK(222) = 19 GOTO 60: PRINT :
  PRINT "FEHLER "; PEEK(222);"!"
26 CLEAR : PRINT : PRINT "BEFEHL (K/C/E)": : CALL 900,B$:
  ON B$ = "" GOTO 26: ON B$ = "K" GOTO 34:
  IF B$ = "E" THEN POKE 216,0: END
28 IF B$ = "C" THEN PRINT CHR$(4);"CATALOG": GOTO 26
30 PRINT CHR$(4);B$: GOTO 26
32 REM PREFIX-WAHL
34 PRINT : PRINT "VON PREFIX/": : CALL 900,P1$:
  ON P1$ = "" GOTO 26:P1$ = "/" + P1$ + "/":
  PRINT "NACH PREFIX/": : CALL 900,P2$:
  ON P2$ = "" GOTO 26:P2$ = "/" + P2$ + "/":
  PRINT CHR$(4);"VERIFY":P2$: PRINT CHR$(4);"VERIFY":P1$
36 REM DIRECTORY-READ
38 DIM C$(100):C = 0: PRINT CHR$(4);"OPEN":P1$;"TDIR":
  PRINT CHR$(4);"READ":P1$: INPUT B$: INPUT B$: INPUT B$
40 C = C + 1: INPUT C$(C): ON C$(C) <> "" GOTO 40:C = C - 1:
  PRINT CHR$(4);"CLOSE": ON C = 0 GOTO 26:
  PRINT CHR$(4);"FRE":CC = 0
42 REM DATEI-WAHL-SCHLEIFE
44 PRINT :CC = CC + 1: ON CC > C GOTO 26:
  PRINT MID$(C$(CC),2,15);" J/N ";
46 GET B$: ON B$ <> "J" AND B$ <> "j" AND B$ <> "N"
  AND B$ <> "n" GOTO 46: PRINT B$;" ";
  ON B$ = "N" OR B$ = "n" GOTO 44:
  B$ = C$(CC):N$ = MID$(B$,2,15)
48 IF RIGHT$(N$,1) = " " THEN
  N$ = LEFT$(N$,LEN(N$)-1): GOTO 48
50 N1$ = P1$ + N$:N1 = LEN(N1$):N2$ = P2$ + N$:
  N2 = LEN(N2$): PRINT N1$;" -> ";N2$:
52 REM DATEI-KOPIE
54 LL = VAL(MID$(B$,65,7)):T$ = ",T" + MID$(B$,18,3):
  IF T$ = "TDIR" THEN PRINT "DIRECTORY!": GOTO 44
56 B = 0:A = 8192:L = 20480: REM $2000+$5000=$7000
58 E = 1: PRINT CHR$(4);"CREATE":N2$:T$: GOTO 62:
  REM E=ERROR
60 E = 0: PRINT CHR$(4);"DELETE":N2$:
  PRINT CHR$(4);"CREATE":N2$:T$
62 IF LL < L THEN L = LL: ON L > 0 GOTO 70: GOTO 44
64 REM SCHUBKOPIE-SCHLEIFE
66 GOSUB 72:B = B + L: ON B + L < LL GOTO 66:L = LL - B:
  IF L = 0 THEN 44
68 REM SETINFO
70 GOSUB 72: POKE 512,N1: FOR X = 1 TO N1:
  POKE 512 + X, ASC(MID$(N1$,X,1)): NEXT:
  POKE 640,N2: FOR X = 1 TO N2:
  POKE 640 + X, ASC(MID$(N2$,X,1)): NEXT:
  CALL 768: GOTO 44
72 PRINT CHR$(4);"BLOAD":N1$:T$;"A":A;"B":B;"L":L:
  PRINT CHR$(4);"BSAVE":N2$:T$;"A":A;"B":B;"L":L:
  RETURN

```

## GETSETINFO

(Als DATA-Statements in KOPY und BATCHKOPY enthalten.  
GETSETINFO funktioniert bei allen PRODOS-Versionen!)

1		ORG	\$0300
2	*		
3	*	Get/Set File-Info	US/16.05.85
4	*		
5	*		
6	*	Altname laden, dessen Parameter	
7	*	{Access/Filetype/Auxtype}	
8	*	zwischen speichern, dann Neunamen	
9	*	laden und mit geänderten	
10	*	Parametern zurückspeichern.	
11	*		
12	DOSWARM	EQU	\$03D0
13	MLI	EQU	\$BF00
14	BELL	EQU	\$FF3A
15	PREYTE	EQU	\$FDDA
16	*		
0300:	20 00 BF	GETINFO1	JSR MLI ;Altname
0303:	C4	HEX	C4
0304:	37 03	DA	CNTA
0306:	B0 26	BCS	ERROR
21	*		
0308:	A9 0A	GETINFO2	LDA #0A ;GET
030A:	8D 49 03	STA	CNTB
030D:	20 00 BF	JSR	MLI ;Neunamen
0310:	C4	HEX	C4
0311:	49 03	DA	CNTB
0313:	B0 19	BCS	ERROR
28	*		



```

0315: A2 03 29 LDX #3
0317: BD 3A 03 30 COPY LDA ACCESSA,X ;Alt nach
031A: 9D 4C 03 31 STA ACCESSB,X ;Neu kopieren
031D: CA 32 DEX
031E: 10 F7 33 BPL COPY
34 *
0320: A9 07 35 SETINFO2 LDA #07 ;SET
0322: 8D 49 03 36 STA CNTB
0325: 20 00 BF 37 JSR MLI ;Neuname
0328: C3 38 HEX C3
0329: 49 03 39 DA CNTB
032B: B0 01 40 BCS ERROR
032D: 60 41 RTS
42 *
032E: 20 DA FD 43 ERROR JSR PRBYTE
0331: 20 3A FF 44 JSR BELL
0334: 4C D0 03 45 JMP DOSWARM
46 *
0337: 0A 47 CNTA HEX 0A ;Nur GET
0338: 00 02 48 NAMEA DA $0200 ;512
033A: 00 49 ACCESSA HEX 00
033B: 00 50 FILETYPEA HEX 00
033C: 00 00 51 AUXTYPEA HEX 0000
033E: 00 52 STORAGEE HEX 00
033F: 00 00 53 BLOCKSA HEX 0000
0341: 00 00 54 MDATEA HEX 0000
0343: 00 00 55 MTIMEA HEX 0000
0345: 00 00 56 CDATEA HEX 0000
0347: 00 00 57 CTIMEA HEX 0000
58 *
0349: 00 59 CNTB HEX 00 ;GET/SET
034A: 80 02 60 NAMEB DA $0280 ;640
034C: 00 61 ACCESSB HEX 00
034D: 00 62 FILETYPEB HEX 00
034E: 00 63 AUXTYPEB HEX 0000
0350: 00 64 STORAGEE HEX 00
0351: 00 65 BLOCKSB HEX 0000
0353: 00 66 MDATEB HEX 0000
0355: 00 67 MTIMEB HEX 0000
0357: 00 68 CDATEB HEX 0000
0359: 00 69 CTIMEB HEX 0000
  
```

91 Bytes

## BILDTEST

(Warnung: BILDTEST funktioniert nur mit BASIC.SYSTEM 1.1!)

```

100 VTAB 24: PRINT : PRINT "BILDTEST": HCOLOR= 7
110 N = 2
120 FOR X = 0 TO N
130 HGR : HPL0T 100,X * 20 TO 200,X * 20:
    REM Waagrecht Strich, unterschiedlich tief
140 PRINT CHR$( 4); "BSAVE BILD, A8192, L8192, B";
    X * 8192
150 PRINT X + 1; " " : NEXT : PRINT
160 FOR X = 0 TO N
170 HGR : PRINT CHR$( 4); "BLOAD BILD, A8192, L8192, B";
    X * 8192
180 PRINT X + 1; " " : NEXT
  
```

## GET FILE INFO (\$C4)

Parameter-Count = \$0A - 1 Byte  
 Name-Pointer - LL HH - 2 Bytes  
 Access - 1 Byte, gepokt  
 File-Type - 1 Byte; gepokt  
 Auxiliary Type - 2 Bytes; gepokt  
 Storage-Type - 1 Byte, gepokt  
 Blocks used - LL HH - 2 Bytes; gepokt  
 Modification-Date - 2 Bytes; gepokt  
 Modification-Time - 2 Bytes; gepokt  
 Creation-Date - 2 Bytes; gepokt  
 Creation-Time - 2 Bytes; gepokt

## SET FILE INFO (\$C3)

Parameter-Count = \$07 - 1 Byte  
 Name-Pointer - LL HH - 2 Bytes  
 Access - 1 Byte  
 File-Type - 1 Byte  
 Auxiliary Type - 2 Bytes  
 Null-Field - 3 Bytes  
 Modification-Date - 2 Bytes  
 Modification-Time - 2 Bytes



<p><b>APPLE -- DISKETTEN LAUFWERKE -- APPLE</b>          Original Disk II m. Controller + DOS 3.3 + Hotbuch DM 905,-          Original Disk II (2 Laufwerke) DM 825,-          Duo-Disk Station Slimline, 2 x 143KB Chignon DM 955,-          Siemens Disk-Laufw. 143KB, m. Kabel + Gehäuse DM 525,-          Abco-Disk-Laufw. Slimline 143KB, m. Kab. + Geh. DM 299,-          Chison Slimline, 143KB, Superleis. m. Kab. + Geh. DM 495,-          Teac 55F, 1 MB un. Kapazität, Shugartbuco DM 599,-          Teac 55F, kpl. in Gehäuse, 40/80 Track, 1MByte DM 648,-          anschlussfertig, mit Umschaltung 40/80 + Kabel DM 495,-          Zweilauferwerk für Apple II C, 143 KB, mit Kabel DM 485,-</p> <p><b>APPLE -- INTERFACES + MAINBOARDS</b>          Disk Controller I, 2 Original o. kompatibel Drives DM 89,-          Super-Controller I, 2x Teac 55F, mit Software DM 109,-          80 Zeich-Karte mit 64K RAM + Softsw. f. Ite DM 189,-          80 Zeich-Karte II, m. Softswitch, 2 Zeichensätze DM 159,-          16K RAM Erweiterung für II und kompatible DM 98,-          2 80A Interface Karte für CPM 2.2 DM 595,-          2 80B Interface Karte für CPM 3.0, m. 64K RAM DM 98,-          Printer-Gratik Interface, Epson kompatibel DM 98,-          Centronic-Parallel Text-Interface Karte DM 98,-          Printer-Gratik Interface, NEC/ITOH kompatibel DM 169,-          Anschluss Kabel I, Parallel + Gratik Interface DM 34,-          Buffer Gratik-Interface, benutzbar für Drucker DM 348,-          Epson/Neoh/Okidata u. andere m. 32K Buffer DM 109,-          Buffer Interface wie vor aber mit 64K Buffer DM 599,-          Anschluss Kabel I, Buffer Interface Karte DM 39,-          232C Serielle Interface Karte DM 119,-          Super Serielle Interface Karte, Full duplex DM 189,-          Sprach (Speech) Karte f. Sprachwiedergabe DM 78,-          6522 Parallel Interface Karte DM 140,-          Clock Karte (Datum/Uhrzeit Ein/Ausgabe DM 129,-          Epson Writer Karte (2716 - 2764) DM 129,-          IEEE-488 Interface Karte DM 398,-          Logo-Karte mit Diskette und Handbuch DM 498,-          Musik Karte m. Diskette und Handbuch DM 119,-          PAL Color Interface Karte (JHF + Video) DM 129,-          RGB Interface Karte (f. Apple II + II+) DM 169,-          Wild Cards (kopier Laser RAM-Bereich) DM 119,-          128K RAM Erweiterungs Karte m. Patchsoftware DM 398,-          256K RAM Erweiterungs Karte m. Patchsoftware DM 498,-          6809 Prozessor Eext. 9 Interfacekarte DM 398,-          IC-Tester Interface Karte (RAMS/TTL 54/74) DM 598,-          Hauptplatine 48K, o. Firmware Eproms, 8 Slots DM 448,-          Hauptplatine 96K, wie vor DM 488,-</p> <p><b>APPLE -- LEERPLATINEN</b>          Leerplatinen der obigen Interface Karte sind alle in vergoldet.          m. Bestück-Druck + Best.-Plan lieferbar.          Leerplatinen mit der Kennzeichnung DM 39,-          Leerplatinen mit der Kennzeichnung DM 24,50          Leerplatinen mit der Kennzeichnung + DM 33,-          Experimentier Platine, für Apple Slot EX300 DM 29,90          Leerplatine Motherboard, 48K, mit Best.-Druck DM 66,-          Leerplatine Motherboard, 64K, mit 6502 + 280 DM 99,-</p>	<p><b>100% Apple-kompatibel</b>          bei Verwendung des          Apple-Datensystems.</p> <p><b>6 Mon. Garantie</b></p> <p><b>Reparaturservice</b></p> <p><b>APPLE - TASTATUREN+LEERGEHÄUSE+NETZTEILE- APPLE</b>          Standard Einbau-Tastatur, ASCII, freibel. 9 Tasten, DM 139,-          Doppelbelg. aller Tasten, Groß+Kleinbuch. Nr. N26 DM 178,-          Tastatur wie vor, jedoch mit 15er Block Nr. N67 DM 249,-          Tastatur Nr. N67, mit sep. Gehäuse, kpl. m. Kabel DM 298,-          Separate Tastatur IBM-Look, anschlussfertig mit DM 399,-          Kabel, Dopp. belg. Tasten, frei prog. Funkt. Tasten DM 98,-          Separate Tastatur, Eprom prog.-bar, Cursorblock DM 119,-          Tastenfeld mit 24 Funktionen, Deutsche o. ASCII DM 399,-          Leergehäuse IBM-Look, f. 2 Laufw. Standard 5 DM 98,-          Leergehäuse wie Mewa 9000, für Einbau von zwei DM 159,-          Slimline-Laufwerken + Tastaturanschluß, Plastic DM 198,-          Leergehäuse wie vor, jedoch in Metall-Ausführung DM 198,-          Leergehäuse IBM-Look, f. 2 Laufw. Standard 5 DM 198,-          Leergehäuse für 1 Slimline Laufwerk, Metall DM 98,-          Leergehäuse f. Duo-Disk = 2x Slimline-Laufwerke DM 179,-          Leergehäuse f. Duo-Disk = 2x Slimline + Netzteil DM 179,-          Schaltteile für APPLE u. kompatible Rechner DM 98,80          +5V/5A -5V/0,5A +12V/2A -12V/0,5A N74 DM 119,80          +5V/7,5A -5V/0,5A +12V/2A -12V/0,5A N75 DM 149,50          +5V/7,5A -5V/0,5A +12V/2A -12V/0,5A N76 DM 149,50          +5V/7,5A -5V/0,5A +12V/2A -12V/0,5A N77 DM 149,50          +5V/7,5A -5V/0,5A +12V/2A -12V/0,5A N78 DM 149,50</p> <p><b>DISKETTEN + DISKETTEN-BOXEN</b>          Profii Disk Box, m. Schloß, Klars-Deckel, 100 Diskets DM 49,-          Disk Box mit Klars-Deckel, ca. 70 Disketten 5 DM 26,-          Hardbox für 10 Disketten, mit Klappdeckel DM 6,90          Disketten, mit Verstärkungsring, 3 Jahre Garantie          ABX 100 = SS/DS 10 Stick, a. 3,98 50 Stick, a. 3,88 500 Stick, a. 3,59          ABX 200 = SS/DS 10 Stick, a. 4,08 50 Stick, a. 3,98 500 Stick, a. 3,69          ABX 400 = DS/DS 10 Stick, a. 4,98 50 Stick, a. 4,88 500 Stick, a. 4,59</p>	<p><b>IBM ... KOMPATIBEL ... IBM ... KOMPATIBEL</b>          Alle Teile unterliegen einer sorgfältigen Endkontrolle und wir übernehmen daher volle Garantie für die Funktionsfähigkeit, sowohl für die gelieferten Leerplatinen, als auch für die fertig bestückten Boards und Interface Karten, die fast ohne Ausnahme mit gesockelten IC's geliefert werden.</p> <p><b>IBM - Kompatible Interface Karten und Mainboards -- IBM</b>          Disk Controller Karte für 2 Disk-Drives *KL-2020 DM 228,-          Color Video Board *KL-2050 DM ab 398,-          Monochrome Video boards *KL-2060 DM ab 328,-          RS 232 Drucker Interface Karte *KL-2074 DM 199,-          Centronics Parallel Interface Karte *KL-2072 DM 140,-          Multifunktions Karte, RAM-Bereich, RS232C, DM 248,-          Parallel, Clock, mit O.K. bestückt *KL-2040 DM 398,-          dto. wie vor, jedoch mit 198K bestückt *KL-2041 DM 548,-          dto. wie vor, jedoch mit 256K bestückt *KL-2042 DM 638,-          512K RAM Karte, mit O.K. bestückt *KL-2055 DM 248,-          dto. wie vor, mit 128K bestückt *KL-2006 DM 398,-          Multifunction Interface Board, RS232C Disk-Controller, Centronics, Clock, Jystick *KL-2021 DM 568,-          und Lightpen-Port *KL-2022 DM 568,-          Eprom Writer Karte (bis 128K benutzbar) *KL-2071 DM ab 990,-          Hardisk Winchester Host-Adapter Karte *KL-2072 DM ab 990,-          Mainboard XT Version, 8 Slots 0 bestückt *KL-1004 DM 699,-          dto. wie vor, jedoch mit 198K bestückt *KL-1005 DM 849,-          dto. wie vor, jedoch mit 256K bestückt *KL-1006 DM 999,-          Mainboard PC Version, 8 Slots, 64K bestückt mit 1 Boot-Eprom, 6 Eprom-Plätze frei *KL-1010 DM 896,-          dto. wie vor, jedoch mit 128K bestückt *KL-1011 DM 948,-          dto. wie vor, jedoch mit 256K bestückt *KL-1012 DM 1298,-          Beiden mit "gekennzeichneten Artikeln gehört zum Lieferumfang ein Manual, und z. T. ein Schaltbild.          Preislisten + Prospekt kostenlos gegen frankierten Rückumschlag.          Alle Leerplatinen werden mit Bestückungsplan geliefert.</p> <p><b>IBM -- Kompatible Leerplatinen + Boards -- IBM</b>          Disk Controller I 2 Drives DM 78,-          Color Gratik Video Board DM 68,-          RS232 Interface Karte DM 68,-          Centronics Parallel Interface Karte DM 68,-          Multifunktions Interface Board DM 78,-          Eprom Writer Interface Karte DM 78,-          Multi I/O Board DM 78,-          Mainboard XT Version, 8 Slots DM 78,-          Mainboard PC Version, 8 Slots DM 90,-          RAM Karte für Speicher-Erweiterung DM 68,-          Alle Leerplatinen werden mit Bestückungsplan geliefert.          Schaltbilder, soweit verfügbar, Lieferung gegen Aufpreis.</p> <p><b>IBM - Gehäuse/Hetzteile/Tastaturen/Diskdrives -- IBM</b>          Rechner Leergehäuse f. Einbau v. 2 Drives DM 198,-          Profi-Tastatur, m. Kabel im Gehäuse, ASCII DM 398,-          dto. wie vor, jedoch deutsche DIN Belegung DM 418,-          Standard Tastatur, IBM-Look, ASCII DM 298,-          Netzteil, mit Ventilator, 100 Watt DM 298,-          dto. wie vor jedoch 135 Watt DM 338,-          Disketten Laufwerk, 2x40 Track DS/360 KB DM 398,-          Harddisk 10MByte, kpl. m. Xbase Controller DM 298,-          Sonderkarte gegen Rückporto, Katalog gegen 3 DM in Briefmarken</p>
--	--	---

**COMPUTER CENTER CONEX**  
 5650 SOLINGEN 11 • Postfach 11 02 06-7 T  
 Telefon (02 12) 7 54 49

**ERICH-WILLI MEYER**  
 6343 FROHNHAUSEN • Postfach 70 11  
 Telefon (02 77 1) 3 50 71

# Graf-quattro

von N.G. Barbieri

## Teil 3: Tricks über Tricks

Was ist eigentlich der Applesoft-Interpreter? Für den Applesoft-Anfänger ein unbekanntes Wesen im Apple, das für Dauerbelästigung mit „SYNTAX ERROR“ sorgt; für den Applesoft-Kenner ein persönlicher Gegner, den es unbedingt auszutricksen gilt; und für den Assemblerprogrammierer schlicht und einfach eine Ansammlung von Maschinenroutinen, die nach Möglichkeit durch eigene Routinen aufzurufen und zu erweitern sind. Dies setzt jedoch genaue Kenntnisse über Aufbau und Struktur des Interpreters voraus.



### BOX.COPY

Damit kommen wir zu der in diesem Beitrag abgedruckten Utility **BOX.COPY**. Wie bereits angekündigt, handelt es sich um das punktweise Kopieren des Inhalts eines in der HGR1-Seite definierten Rechtecks auf eine andere Stelle derselben

Seite oder wahlweise auf die HGR2-Seite. Bindet man diese Utility in die Routinen der vergangenen Beiträge ein, dann ist es von Applesoft aus sehr einfach: Cursoren einmal positionieren, dann ein zweites Mal, ein POKE 255,0 für Übertragung HGR1-HGR1 oder POKE 255,1 für Übertragung HGR1-HGR2, ein CALL 25297, und die Sache läuft schon. Will man nur diese Routine allein in eigene Programme einbauen, dann immer – in der Reihenfolge X1LOW, X1HIGH, Y1 und X2LOW, X2HIGH, Y2 – die Eckwerte der ursprünglichen Box in \$03C4 (964) bis \$03C9 (969) und die Eckwerte des Zielgebietes in \$03CA (970) bis \$03CF (975) poken, Seitenflag in \$00FF (255) nicht vergessen und nun **CALL 25297**.

Zu dem eigentlichen Assemblerprogramm, insbesondere für diejenigen, die in der Maschinensprache ihre ersten Schritte wagen, ist noch einiges zu sagen: Es handelt sich um eine reine „Bit-an/Bit-aus-Übertragung“ ohne Rücksicht auf das letzte Bit oder die Positionierung (d.h. Color-Bereich oder gerade und ungerade Hires-Spalte), so daß beim Übertragen von farbigen Ausschnitten komische Farbverschiebungen eintreten können. Demgegenüber klappt es bei schwarzweiß immer.

Das Programm besteht aus zwei Teilen: Der erste Teil untersucht die relative Position der zwei Rechtecke und entscheidet,

ob das Kopieren von links nach rechts und von oben nach unten oder umgekehrt verlaufen soll, da es für den Fall, daß sich Ursprungs- und Zielbox überschneiden, passieren könnte, daß man kopiert, was schon einmal kopiert worden ist. Das Problem ist ähnlich wie beim Moven von Speicherbereichen, nur daß hier die Überschneidung zweidimensional sein könnte. Der zweite Teil (ab LOOP) holt sich einfach einen Punkt von einer Stelle und bringt ihn an einer anderen Stelle unter, bis der ganze Inhalt der Box übertragen worden ist (siehe auch Kommentar).

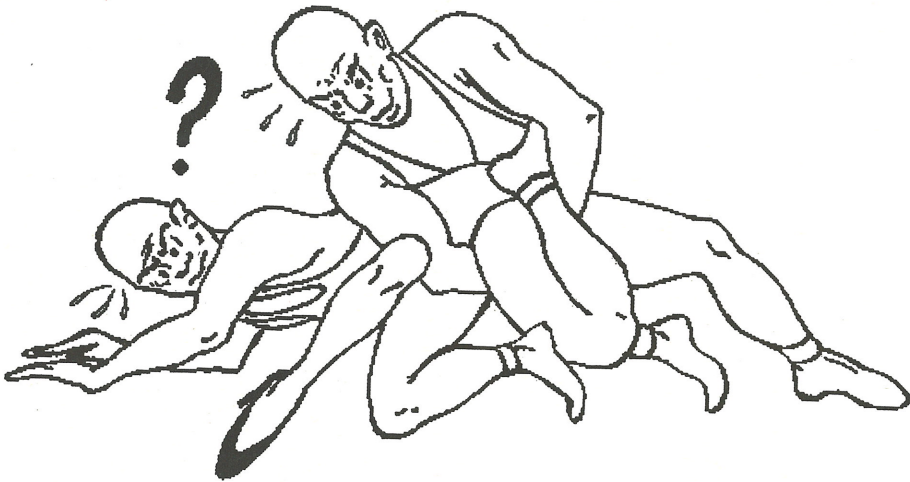


### Drei Tricks für Anfänger

#### 16-Bit-Vergleich

Jeder kennt die CMP-Funktion (Compare) zum Verzweigen (Branch) je nach Carry- oder Zero-Flag, obwohl gelegentlich beim Bestimmen, was größer, gleich oder kleiner





IBM-PC/XT-COMPATIBLE COMPUTER-ZUBEHÖR	
Mainboard XT 1-Board (bis 256K auf) + 16K Speicherleiste	95,-
Disk Controller (ser. f. 4 x 360K-Drives)	49,-
Monochrome Graph. f. par. Printer-Leiste (Hercules comp.)	53,-
Color RGB + Video-Graphik-Leiste	65,-
Parallel-Printer-Leiste	70,-
512K-RAM Leiste	70,-
Multi I/O Leiste (Printer, Ser., Game, Uhr gepuffert)	89,-
384K-Multi I/O Leiste (bis 256K auf) + 8SI best + gepr.	
Dr. + Boot-From	696,-
Mainboard wo jedoch 256K-vollbest.	777,-
512K-RAM Onp. Aufbauplatz (3) einzeln gepuffert!	239,-
Disk Controller (par. f. 4 x 360K-Drives)	245,-
Monochrome Graph. f. par. Printer-Karte (Hercules comp.)	258,-
Color RGB + Video-Graphik-Karte	268,-
Parallel-Printer-Karte	132,-
512K-RAM-Karte (par. gepuffert) (Extra Kabel = 75,-)	150,-
384K-Multi I/O Karte, RS 232, par. Printer, Intert. Game/JoyStick Port	154,-
Real Time Clock gepuffert, Dr. + RAM-Disk-Copier zur Show	145,-
384K-Multi I/O wo vollbest + gepr.	759,-
Multi I/O Karte (Par. Ser. Uhr gepuffert) gepr.	450,-
Prof. Funktions-tastat. dr. oder ASCII + 15er Block	347,-
320K Speicher im eingetauscht. 16er-Block/Konsole/vollbest	498,-
380K-Diskdrive Slimline ohne Gehäuse	2.399,-
XT 256K-RAM Hoch Intel. Intel. Boot-From. f. 360K-Drive + Contr.	2.545,-
f. f. Tastatur im IBM-Hook, like Geh. f. 150W-Netzteil best + gepr.	2.645,-
XT 256K wo + Color-RGB + Video-Graphik-Karte	2.845,-
XT 256K wo aber mit Monochrome-Graphik-Karte + par. Printer int.	2.929,-
IBM-PC-Systemeinheit + Laufwerk	3.719,-
IBM-Bildschirm Color	698,-
PC 18-Kommode ab Tast.	1.698,-
GENIE 16 G (256K-RAM bis 640K auf) 2 x 360K-Drive,	4475,-
an + par. Printer + incl. d. Tastatur	4895,-
Schwab + Manual für IBM f. compatible Geräte	
Microsoft flight simulator	145,-
Lotus 1-2-3	1095,-
Emul.	1195,-
D-base III	1395,-
UCSD P-System	1595,-
Symphony	1495,-

Apple-BUS-COMPUTER-LINIEN+PERIPHERIE	
Motherboard durchkontaktiert mit Lötplace	66,-
M-board II 48K, gepackelt, vollbest + gepr.	429,-
M-board II 64K+280 CPU, gepackelt, v.best + gepr.	524,-
M-board IIe 64K vollbest+gepr.	749,-
APOLLO II Konsole/Lesegeräte	275,-
APOLLO II Gehäuse incl. Funkt.-Tast.	275,-
Leertaste: 16K-RAM, 280CPU, Controller DOS 3.3,	
Parallel-Drucker-Karte, Ser. Int. V24, 802/242-Karte	
PAL-Modul-Karte	18,95
Entwicklungs-Charger-Leertaste durchkontaktiert	28,50
16K-Speicher-Karte gepuffert	85,-
APOLLO -orig. 16K-Language Karte gepr.	129,-
230 CPU-Karte gepuffert	77,-
Controller DOS 3.3-Karte gepuffert	77,-
Auto-Controller DOS 3.3/3-Karte gepr.	145,-
Epoch ARD 2 Controller + autom. Softw. + Handbuch	268,-
Parallel-Drucker-Karte	95,-
80bit par. Gruppen-Int. + 84K-Buffer-Kabel gepr.	99,-
Seriell-Interface-Karte V24 gepuffert	99,-
Super-Seriell-Interface Karte gepr.	169,-
80-Zeichen/24-Zellen-Karte gepuffert	119,-
802/242-Karte+Schwabe-Schalter Leertaste	45,-
802/242-Karte+Schwabe-Schalter	145,-
IEEE-488 Int. Karte gepr.	375,-
PAL-Modul/Color-Karte gepuffert	95,-
PAL-RGB-Color Karte	159,-
128K-Speicher-Leertaste	59,-
128K-Speicher-Karte+Software+Manuel	99,-
128K-Speicher-Karte+Software gepuffert	369,-
256K-Speicher-Leert+4 Spezial (Cs)+Software	195,-
256K-Speicher-Karte+Software gepuffert	495,-
Maus incl. Software	189,-
EPROM-Burner (2706/16/32/64) gepuffert	119,-
Modern Eurocom-Karte gepr. + Software + Man.	495,-
UHF-TV-Modulator universell	27,50
486	
Datashop 212 A-Kausalkopier-FTZ	288,-
APOLLO II (48K) mit f. 15er-Block/vollbest+gepr.+F.T.	1650,-
APOLLO // (512K) mit f. 15er-Block/vollbest+gepr.+F.T.	1999,-
APOLLO // (48K) mit f. 15er-Block/vollbest+gepr.+F.T.	899,-
APOLLO // (48K) mit f. 15er-Block/vollbest+gepr.+F.T.	1475,-
APOLLO // (48K) mit f. 15er-Block/vollbest+gepr.+F.T.	987,-
APOLLO // (48K) mit f. 15er-Block/vollbest+gepr.+F.T.	1439,-
APOLLO // (48K) mit f. 15er-Block/vollbest+gepr.+F.T.	1310,-
Aufbau für M-board // (48K) 280 CPU	95,-
APOLLO // (48K) mit f. 15er-Block/vollbest+gepr.+F.T.	1439,-
APOLLO // (48K) mit f. 15er-Block/vollbest+gepr.+F.T.	1310,-
APOLLO // (48K) mit f. 15er-Block/vollbest+gepr.+F.T.	777,-
APOLLO // (48K) mit f. 15er-Block/vollbest+gepr.+F.T.	667,-
APOLLO // (48K) mit f. 15er-Block/vollbest+gepr.+F.T.	426,-
APOLLO // (48K) mit f. 15er-Block/vollbest+gepr.+F.T.	387,-
Epoch Disk 1,2 Mbytes incl. Netdisk+Contr.	1099,-
Maus f. Apple // incl. Manual+Software	188,-
TIOS	79,-
APPLESORT-Tutoriel-Reference-Manual engl.	109,-
APPLE-DOS 3.3 Manual engl.	48,-
APPLE-DOS 3.3 Hib. dt.	69,-
APPLE-DOS 3.3 Disk engl.	69,-
APPLE-Pascal Reference-Manual engl.	48,-
APPLE-Pascal (V.2.2) 1. Disk + Manual engl.	179,-
APPLE-Pascal Reference Hib. dt.	69,-
APPLE-Pascal Language Hib. dt.	69,-
APPLE-PORTH Manual engl.	69,-
APPLE-Forman Manual engl.	69,-
CPI-Software V.0.1 Manual engl.	69,-
APPLE // (Anwendungsbeispiel) PAL-Karte	48,-
6502-Programmier-Handbuch deutsch.	39,-
6502-Programmier-Handbuch englisch (Bridg Text)	39,-
Software + Manual f. APPLE-compatible Geräte	
apple how to apple comp. soft. manual, chinese.	
apple II ver. 2, 4, wordstar + mailmerge + spellstar mit installierprogram.	
apple logo, detector, calculator, plot file, plot graph, plot report, apple	
fontan, cp/m c language, text simulator, c programming language,	
flight simulator, visicalc, vischedule, visilisting, multiple, superword,	
microsoft cobol 60, microsoft fortran, msd, graphics-erase utility.	
APOLLO // (64K) PAL + 15er-Block, vollbest. und gepr.	1045,-
APOLLO // (64K) PAL + 15er-Block, vollbest. und gepr.	2199,-
APOLLO // (64K) PAL + 15er-Block, vollbest. und gepr.	3110,-
802/242-Karte+Schwabe-Karte gepuffert	179,-
APPLE IIc (128K) dt. Tastatur	2495,-
280 Karte gepr. f. APOLLO-RAM-Disk-Software hb.	495,-
Macintosh System (128K)	6945,-
Macintosh System (512K)	9995,-
Macintosh Umrichter von 128K auf 512K	899,-
LSAOV-DISK-Computer-Mod-PC dt. Tastatur	
EPSON LX 501 + NEB	249,-
EPSON RX 80 +, 8bit/parallel	869,-
EPSON RX 807F +, 8bit/parallel	1019,-
EPSON RX 100 +, 8bit/parallel	1269,-
EPSON RX 807F +, 8bit/parallel	1329,-
EPSON FX 100 +, 8bit/parallel	1739,-
EPSON LX 100 +, 8bit/parallel	2299,-
EPSON Traktorersatz für FX 80	129,50
APPLE-EPSON-Drucker-Graphic-Interface-Kabel	129,50
IBM SVL55, FX18/80 Superdisk-Kassette	15,95
Brother Typendrucker, HR 15XL 80/par. neues Mod.	1389,-
STAR 8610, CD + 15er-Block/vollbest+gepr.	1299,-
Zenith ZVM 123 gr. 12" oder Zenith ZVM 122 or. 12" 15 MHz	277,-
SANYO DM 2312 12" senig 15 MHz	265,-
SANYO 8112 Monochrome Monitor 12" gr.	587,50
Monochrome 12" Monitor entspiegel. gr. or. 22 MHz	389,-

ner ist, Verwirrungen entstehen können. Belegen aber die zu vergleichenden Werte zwei Bytes (wie im Falle der X-Koordinaten), ist die einfachere Version erst ein Vergleich der High-Bytes und, wenn sie gleich sind, ein zweiter Vergleich der Low-Bytes, um die Sache endgültig klärzuzumachen. Der im Programm gezeigte Weg (Trick Nr. 1) ist kürzer und schneller. Erst ein Vergleich der zwei Low-Bytes, um das Carry-Flag entsprechend zu setzen, dann eine Subtraktion (mit Carry) der zwei High-Bytes, so daß am Ende die üblichen Branches erfolgen können.

### 16-Bit-Dekret

Was beim 16-Bit-Inkrement klappt, d.h. das High-Byte wird dann inkrementiert, wenn das Low-Byte den Wert \$00 einnimmt (ansonsten BEQ), ist beim Dekrementieren nicht gegeben, da der Wert \$00 des Low-Bytes auch mitgenommen werden muß. Also greift man normalerweise zur 16-Bit-Subtraktion, wobei vom High-Byte immer eine 0 mit Carry subtrahiert wird. Mit dem kleinen Kniff (siehe Trick Nr. 2) wird beim 16-Bit-Dekret auch die 0 immer mitgenommen.

### HSCRN-Routine

An sich hatte ich ursprünglich vor, diese Routine als separate Utility zur allgemeinen Benutzung zu gestalten, aber dann habe ich es mir anders überlegt. Trotz ausgedehntem Einsatz von Zeropage-Pointern ist das bitweise Übertragen auch so langsam genug! Überträgt man eine Box in der Größe der ganzen Grafikseite, d.h. 53760 Punkte, dauert dieses etwa 33 Sekunden. Eine gesonderte Routine hätte

für jeden einzelnen Punkt zusätzlich ein JSR und ein RTS gekostet, also ganze 12 Zyklen mehr, und das macht sich bemerkbar!

Ich möchte dem Leser aber eine separate HSCRN-Routine nicht vorenthalten. Die Routine **HSCRN** ist völlig relokativ, so daß man wie folgt vorgehen kann:

Ein BLOAD HSCRN, A... an jede beliebige Stelle. Danach die in Frage kommenden Koordinaten in der üblichen Reihenfolge (XLOW, XHIGH und Y) in \$0000, \$0001 und \$0002 poken, dann CALL... (Stelle von A = Anfangsadresse der Routine), danach ein N = PEEK (3). Ist N = 0, dann ist der angepeilte Punkt schwarz, ist N < > 0, entsprechend weiß. Einfach, nicht wahr?

Mit dieser Folge sind die Assembler-Routinen für das erste Modul der Serie Grafikquattro komplett. Jetzt heißt es zusammenstellen, also Diskette mit allen Assembler-Routinen in das Laufwerk legen und dann:

```
BLOAD XPLOT
BLOAD CURSOR 1
usw. bis
```

```
BLOAD BOX.COPY
BSAVE GRAF.QUATTRO1, A$6000, L$466
```

Wer sich diese Mühe nicht machen will, kann auch die entsprechende Peeker-Diskette bestellen!

Eigentlich könnte jetzt jeder Applesoft-Programmierer seinen eigenen Hires-Page-Editor selber schreiben. Nichtsdestoweniger wird mein eigener Editor demnächst im Peeker erscheinen.

### Disketten in Box + Aufkleber I. Wahl 10 St./100 St. à

5 1/4" Verstärkt	BASF	3M	SCOTCH	Fuji	DATA	MAGNE.	NEUTRAL
1X, SS/SD	4,19 / 3,99						3,55/3,28
1D, SS/DD	4,29 / 4,0	6,88 / 4,78	4,57 / 4,37	3,96/3,76	3,26/2,96		
2D, DS/DD	5,99 / 5,79	6,88 / 6,88	5,87 / 5,67		4,74/5,55		
1X, HT	5,79 / 5,59						
2D, 80/10	7,69 / 7,49	6,98 / 6,28	7,97 / 7,77				
2D/17MB	13,89 / 12,89		14,87 / 13,87				
1X, Hard	4,49 / 4,29						
8"							
1X, SS/SD	4,49 / 4,29	6,68 / 6,48	6,97 / 6,67				
1D, SS/DD	4,79 / 4,59	6,88 / 6,28					
2D, SS/DD	7,09 / 6,89	6,88 / 6,28	10,17 / 9,77				
1X, Hard	4,49 / 4,29	6,68 / 6,48					
3 1/2"							
Einselig	10,99 / 9,99	11,78/11,38					
Zweiselig	14,99/14,29						
3M Scotch 5 1/4" Plus-III-Reinigungsset (2 Disk.) einseitig							48,95
BASF 5 1/4" Reinigungsplatte einseitig							48,50
5 1/4" Disketten-Archivbox für 15 Disk. 3M							5,75/5,49,95
5 1/4" Disketten-Archivbox für 15 Disk. IBM							19,95
5 1/4" Disketten-Archivbox für 15 Disk. IBM							26,50
5 1/4" Disketten-Archivbox für 15 Disk. IBM							39,95
5 1/4" Disk-K-K. Rauchschild abschreibbar, f. ca. 80-100 Disk.							47,50
4000 B. Floppydisk (5 1/4") 128Kbit, neu beschreibbar							66,50
4000 E. Aufkl. Doppel (107 x 38 mm auf 240 x 12", perf. Träger)							249,-
4000 E. Aufkl. einzeln (107 x 38 mm auf 125 x 12", perf. Träger)							159,-

### OSZILLOSCOPE HAMEG ab Lager

Bei Vorauszahlung frei Emslangung unverpackt in der BRD ausgenommen Passer und Einheiten, sonst N.A. v.5 ab DM 50,-

Öffnungszeiten: Mo. Di. Do. Fr. 10-18 Uhr, Mi. Sa. v. 10-14 Uhr, So. v. 10-19 Uhr, Tel. Best.: Mo. Di. Do. Fr. von 10-19 Uhr, Mi. u. Sa. wie Öffnungszeiten

GEWÄHRLEISTUNG: 6 Monate, auf alle bei uns gekauften Geräte, durch unsere eigene Service-Abteilung

REPARATUREN: an Apple- u. Compatible Geräten + Zubehör führt unser Spezialistenamt garantiert zuverlässig + besonders Kostengünstig aus. Sprechen Sie mit uns.



## BOX.COPY

```

1          ORG $62D1      ;25297
2          *
3          * BOX.COPY
4          *
5          *
6          * von N.G. Barbieri/1985
7          *
8          * Aufrufen mit CALL 25297
9          *
10         * Kopiert den Inhalt einer Box
11         * bitweise auf eine andere
12         * Stelle derselben (HGR1) oder
13         * der anderen (HGR2) Grafik-
14         * seite.
15         *
16         * Applesoft-Routinen u. -Pointer
17         *
18         HPOSN EQU $F411
19         HPLOT0 EQU $F457
20         GBASL EQU $26
21         HMASK EQU $30
22         HCOLORZ EQU $E4
23         HPAG EQU $E6
24         *
25         * Cursorposition der
26         * ursprünglichen Box
27         *
28         CALXL EQU $3C4
29         CALXH EQU $3C5
30         CALY EQU $3C6
31         CA2XL EQU $3C7
32         CA2XH EQU $3C8
33         CA2Y EQU $3C9
34         *
35         * Cursorposition
36         * des neuen Platzes
37         *
38         CULXL EQU $3CA
39         CULXH EQU $3CB
40         CULY EQU $3CC
41         CU2XL EQU $3CD
42         CU2XH EQU $3CE
43         CU2Y EQU $3CF
44         *
45         * Zähler für X- u. Y-Richtungen
46         *
47         COUNXL EQU $1D
48         COUNXH EQU $1E
49         COUNY EQU $1F
50         *
51         * X,Y-Koordinaten des zu
52         * übertragenden Bits
53         *
54         XLOW EQU $00
55         XHIGH EQU $01
56         Y EQU $02
57         *
58         * Links/Rechts-Flag
59         *
60         XFLAG EQU $03
61         *
62         * X,Y-Zielkoordinaten
63         *
64         XLOW2 EQU $06
65         XHIGH2 EQU $07
66         Y2 EQU $08
67         *
68         * Oben/unten-Flag
69         *
70         YFLAG EQU $09
71         *
72         * Temporäre Ablage für beide
73         * X-Koordinaten u. X-Zähler
74         *
75         TEXL1 EQU $F9
76         TEXH1 EQU $FA
77         TEXL2 EQU $FB
78         TEXH2 EQU $FC
79         COXL EQU $FD
80         COXH EQU $FE
81         *
82         * Grafikseite-Übertragungsflag:
83         * wenn 0 auf HGR1, sonst auf HGR2
84         *
85         PFLAG EQU $FF
86         *
87         *

```

```

62D1: AD C6 03 88          LDA CA1Y
62D4: CD CC 03 89          CMP CULY
62D7: D0 06 90          BNE STEP1
91          *
92          * Wenn beide Y gleich, dann Flag
93          * auf 2!
94          *
62D9: A9 02 95          LDA #$2
62DB: 85 09 96          STA YFLAG
62DD: D0 0C 97          BNE TESTX
98          *
99          * Feststellen, ob von oben nach
100         * unten oder umgekehrt und YFLAG
101         * entsprechend setzen.
102         *
62DF: B0 06 103         STEP1 BCS DOINC
62E1: A9 01 104         LDA #$1
62E3: 85 09 105         STA YFLAG
62E5: D0 04 106         BNE TESTX
62E7: A9 00 107         DOINC LDA #$0
62E9: 85 09 108         STA YFLAG
109         *
110         * Wenn Übertragung auf der an-
111         * deren Seite, dann egal
112         * welche Richtung!
113         *
62EB: A5 FF 114         TESTX LDA PFLAG
62ED: D0 27 115         BNE DOXINC
116         *
117         * Feststellen, ob von links nach
118         * rechts oder umgekehrt und XFLAG
119         * entsprechend setzen.
120         *
62EF: AD C5 03 121         LDA CALXH
62F2: CD CB 03 122         CMP CULXH
62F5: F0 04 123         BEQ TXLOW
62F7: B0 1D 124         BCS DOXINC
62F9: 90 15 125         BCC DOXDEC
62FB: AD C4 03 126         TXLOW LDA CALXL
62FE: CD CA 03 127         CMP CULXL
6301: D0 0B 128         BNE STEP2
129         *
130         * Wenn gleiche X- und Y-Werte u.
131         * gleiche Seite, warum übertragen?
132         * Jetzt wird klar, warum bei
133         * gleichen Y-Werten das YFLAG
134         * auf 2 steht!
135         *
6303: A5 09 136         LDA YFLAG
6305: C9 02 137         CMP #$2
6307: F0 19 138         BEQ RET1
139         *
6309: 85 03 140         STA XFLAG
630B: 4C 1A 63 141         JMP STEP3
630E: B0 06 142         STEP2 BCS DOXINC
6310: A9 01 143         DOXDEC LDA #$1
6312: 85 03 144         STA XFLAG
6314: D0 04 145         BNE STEP3
6316: A9 00 146         DOXINC LDA #$0
6318: 85 03 147         STA XFLAG
148         *
631A: AD C6 03 149         STEP3 LDA CA1Y
631D: CD C9 03 150         CMP CA2Y
6320: D0 05 151         BNE DOIT
152         *
6322: A9 20 153         RET1 LDA #$20
6324: 85 E6 154         STA HPAG
6326: 60 155         RTS
156         *
157         * Für den Y-Zähler testen, was
158         * wovon subtrahiert werden
159         * muß!
160         *
6327: 90 0C 161         DOIT BCC SUBY2
6329: 38 162         SEC
632A: ED C9 03 163         SBC CA2Y
632D: 85 1F 164         STA COUNY
632F: A5 09 165         LDA YFLAG
6331: F0 0F 166         BEQ MOD1
6333: D0 16 167         BNE MOD2
168         *
6335: 38 169         SUBY2 SEC
6336: AD C9 03 170         LDA CA2Y
6339: ED C6 03 171         SBC CA1Y
633C: 85 1F 172         STA COUNY
633E: A5 09 173         LDA YFLAG
6340: F0 09 174         BEQ MOD2
175         *
176         * Y-Startpunkte setzen.

```

## Für Ihre Unterlagen

Abonnement bestellt

am: \_\_\_\_\_

### Vertrauensgarantie:

Ich habe davon Kenntnis genommen, daß ich die Bestellung schriftlich durch Mitteilung an den Dr. Alfred Hüthig Verlag, Postfach 102869, 6900 Heidelberg 1 innerhalb von 7 Tagen widerrufen kann. Zur Fristwahrung genügt die rechtzeitige Abendung des Widerrufs (Datum des Poststempels).

### peeker

Leserservice

Postfach 102869

6900 Heidelberg 1

## Für Ihre Unterlagen

Folgende Bücher bestellt:

\_\_\_\_\_

\_\_\_\_\_

\_\_\_\_\_

\_\_\_\_\_

\_\_\_\_\_

am: \_\_\_\_\_

bei: \_\_\_\_\_

### peeker

Versandbuchhandlung

Postfach 102869

6900 Heidelberg 1

## Für Ihre Unterlagen

Folgende Disketten  
und Programme bestellt:

\_\_\_\_\_

\_\_\_\_\_

\_\_\_\_\_

\_\_\_\_\_

\_\_\_\_\_

am: \_\_\_\_\_

bei: \_\_\_\_\_

### peeker

Softwareabteilung

Postfach 102869

6900 Heidelberg 1



## Abo-Karte

Ja, ich möchte **peeker** abonnieren.

Liefere Sie mir **peeker** ab Ausgabe ..... (1985 erscheinen 11 Ausgaben – 1 Doppelnummer) zum Jahresbezugspreis von DM 72,- (Inland) incl. MwSt. Die Lieferung erfolgt frei Haus. Porto, Verpackung und Zustellgebühren übernimmt der Verlag. Der Jahresbezugspreis für das Ausland beträgt DM 72,- incl. MwSt., zzgl. DM 16,80 Versandkosten.

Ich wünsche jährliche Berechnung durch:

- Verlagsrechnung       Abbuchung von meinem Bank- bzw. Postscheckkonto

Bank / PschA \_\_\_\_\_

Bankleitzahl \_\_\_\_\_ Kto.-Nr. \_\_\_\_\_

Datum \_\_\_\_\_ Unterschrift \_\_\_\_\_



## Buch-Shop

Bitte senden Sie mir gegen Rechnung folgende Bücher:

Menge	Autor, Titel	à DM	gesamt DM

Datum \_\_\_\_\_ Unterschrift \_\_\_\_\_



## Software-Karte

Bitte senden Sie mir  
gegen Rechnung folgende Apple-Programme:

- |  |  |
|--|--|
| <input type="checkbox"/> Peeker-Sammdiskette, einzeln<br>Disk# _____, Disk# _____<br>Disk# _____, Disk# _____<br>Preis je Disk DM 28,- (einzeln) | <input type="checkbox"/> Apple DOS 3.3, Begleitdiskette, DM 28,-           |
| <input type="checkbox"/> Peeker Sammdiskette,<br>im Fortsetzungsbezug<br>ab Disk # _____<br>(Mindestbezug 6 Disketten)<br>Preis je Disk DM 20,-  | <input type="checkbox"/> Apple ProDOS, Band 1, Begleitdiskette,<br>DM 28,- |
|  | <input type="checkbox"/> Apple ProDOS, Band 2, Begleitdiskette,<br>DM 28,- |
|  | <input type="checkbox"/> Apple Assembler, Begleitdiskette, DM 28,-         |
|  | <input type="checkbox"/> ProDOS-Editor 1.0, Programm, DM 98,-              |
|  | <input type="checkbox"/> MMU 2.0, Programm, DM 98,-                        |
|  | <input type="checkbox"/> INPUT 2.0, Programm, DM 98,-                      |
|  | <input type="checkbox"/> Softbreaker 1.0, Programm, DM 48,-                |
|  | <input type="checkbox"/> DB-Meister, Programm, DM 290,-                    |
|  | <input type="checkbox"/> Superplot, Programm, DM 48,-                      |
|  | <input type="checkbox"/> Superquick, Programm, DM 48,-                     |

Datum \_\_\_\_\_ Unterschrift \_\_\_\_\_





## Abo-Karte

Name \_\_\_\_\_

Firma \_\_\_\_\_

Abteilung \_\_\_\_\_

Straße \_\_\_\_\_

PLZ/Ort \_\_\_\_\_

**Vertrauensgarantie:**  
Ich habe davon Kenntnis genommen, daß ich die Bestellung schriftlich durch Mitteilung an den Dr. Alfred Hüthig Verlag, Postfach 10 28 69, 6900 Heidelberg 1 innerhalb von 7 Tagen widerrufen kann. Zur Fristwahrung genügt die rechtzeitige Absendung des Widerrufs (Datum des Poststempels).

Datum \_\_\_\_\_

Unterschrift \_\_\_\_\_

**Verlagshinweis:**  
Das Abonnement verlängert sich zu den jeweils gültigen Bedingungen um ein Jahr, wenn es nicht 2 Monate vor Jahresende schriftlich gekündigt wird.



## Buch-Shop

Karte bitte vollständig ausfüllen

Vorname, Name \_\_\_\_\_

Firma \_\_\_\_\_

Straße \_\_\_\_\_

PLZ/Ort \_\_\_\_\_

Telefon mit Vorwahl \_\_\_\_\_



## Software-Karte

Karte bitte vollständig ausfüllen

Vorname, Name \_\_\_\_\_

Firma \_\_\_\_\_

Straße \_\_\_\_\_

PLZ/Ort \_\_\_\_\_

Telefon mit Vorwahl \_\_\_\_\_

POSTKARTE

**peeker**  
Leserservice

Postfach 10 28 69

6900 Heidelberg 1

POSTKARTE

**peeker**  
Versandbuchhandlung

Postfach 10 28 69

6900 Heidelberg 1

POSTKARTE

**peeker**  
Softwareabteilung

Postfach 10 28 69

6900 Heidelberg 1

## INPUT 2.0

Ein Bildschirm-  
Maskengenerator  
für DOS 3.3 und ProDOS  
von U. Stiehl

1984, Diskette und Manual, DM 98,-  
ISBN 3-7785-1021-5

„Input 2.0“ liegt wahlweise in der Bank 1 oder Bank 2 der Language Card und wird durch einen kurzen Driver in den unteren 48K aufgerufen.

Für jedes Feld der Bildschirmmaske lassen sich u. a. definieren: Feldlänge (bis zu 255 Zeichen) – Vtab – Htab – Datentyp (insgesamt 8 Typen) – Scrollflag (starre oder dynamische Maske) – Ctrllflag – Füllflag – Löschflag – Bildschirmflag (40- oder 80-Z-Darstellung). Innerhalb eines Eingabefeldes besteht jeder denkbare Redigierkomfort (Insert, Delete, Rubout, Restore usw.).

Gerätevoraussetzung: Apple IIe oder IIc; ferner Apple II+ im 40-Zeichenmodus

## MMU 2.0 Memory Managements Utilities

für die Apple IIe 64K-Karte  
DOS 3.3 (und ProDOS)

von U. Stiehl

1984, Diskette und Manual, DM 98,-  
ISBN 3-7787-1023-1

Insgesamt enthält die neue „MMU 2.0“-Diskette über 25 Programme, die neue Einsatzmöglichkeiten für die Extended 80 Column Card (erweiterte 80-Z-Karte = 64K-Karte für den Apple IIe) erschließen. Ein Teil der Programme laufen auch auf dem Apple II Plus, doch ist „MMU 2.0“ primär für 64K-Karte-Besitzer gedacht.

Gerätevoraussetzung: Apple IIe mit 64K-Karte oder IIc

## Softbreaker 1.0

Eine softwaremäßige Interrupt-Utility  
für die Apple IIe 64K-Karte

von U. Stiehl

1984, Diskette und Manual, DM 48,-  
ISBN 3-7785-1022-3

Softbreaker ist ein Assemblerprogramm, mit dessen Hilfe Programme, die sich von der 64K-Karte (= Extended 80 Column Card für den Apple IIe) starten lassen, unterbrochen, gespeichert, geladen und exakt an der Stelle der Unterbrechung fortgeführt werden können. Dadurch ist es auch möglich, Sicherungskopien von sogenannten kopiergeschützten Programmen herzustellen.

Mit Softbreaker unterbrochene Programme werden komplett, d. h. die ganzen 64K einschließlich Language Card, in nur ca. 11 Sekunden auf einer formatierten Diskette gespeichert.

Gerätevoraussetzung: Apple IIe mit 64K-Karte

**Hüthig Software Service,  
Postfach 10 28 69, D-6900 Heidelberg**



```

177 *
6342: AE C9 03 178 MOD1 LDX CA2Y
6345: AC CF 03 179 LDY CU2Y
6348: 4C 51 63 180 JMP STEP4
181 *
634B: AE C6 03 182 MOD2 LDX CA1Y
634E: AC CC 03 183 LDY CU1Y
184 *
6351: 86 02 185 STEP4 STX Y
6353: 84 08 186 STY Y2
6355: E6 1F 187 INC COUNY
188 *
189 * Für den X-Zähler testen, was
190 * wovon subtrahiert werden
191 * muß!
192 *
193 * *** TRICK NR. 1 ***
194 *
6357: AD CA 03 195 LDA CU1XL
635A: CD CD 03 196 CMP CU2XL
635D: AD CB 03 197 LDA CU1XH
6360: ED CE 03 198 SBC CU2XH
6363: B0 17 199 BCS SUBX2
200 *
6365: 38 201 SEC
6366: AD CD 03 202 LDA CU2XL
6369: ED CA 03 203 SBC CU1XL
636C: 85 1D 204 STA COUNXL
636E: AD CE 03 205 LDA CU2XH
6371: ED CB 03 206 SBC CU1XH
6374: 85 1E 207 STA COUNXH
6376: A5 03 208 LDA XFLAG
6378: D0 17 209 BNE MOD3
637A: F0 28 210 BEQ MOD4
211 *
637C: 38 212 SUBX2 SEC
637D: AD CA 03 213 LDA CU1XL
6380: ED CD 03 214 SBC CU2XL
6383: 85 1D 215 STA COUNXL
6385: AD CB 03 216 LDA CU1XH
6388: ED CE 03 217 SBC CU2XH
638B: 85 1E 218 STA COUNXH
638D: A5 03 219 LDA XFLAG
638F: D0 13 220 BNE MOD4
221 *
222 * X-Startpunkte setzen.
223 *
6391: AE C7 03 224 MOD3 LDX CA2XL
6394: AC C8 03 225 LDY CA2XH
6397: 86 00 226 STX XLOW
6399: 84 01 227 STY XHIGH
639B: AE CD 03 228 LDX CU2XL
639E: AC CE 03 229 LDY CU2XH
63A1: 4C B4 63 230 JMP REST
231 *
63A4: AE C4 03 232 MOD4 LDX CA1XL
63A7: AC C5 03 233 LDY CA1XH
63AA: 86 00 234 STX XLOW
63AC: 84 01 235 STY XHIGH
63AE: AE CA 03 236 LDX CU1XL
63B1: AC CB 03 237 LDY CU1XH
238 *
63B4: 86 06 239 REST STX XLOW2
63B6: 84 07 240 STY XHIGH2
63B8: E6 1D 241 INC COUNXL
63BA: D0 02 242 BNE WEITER
63BC: E6 1E 243 INC COUNXH
244 *
245 * X-Koordinaten und X-Zähler
246 * in die Ablage bringen.
247 *
63BE: A6 00 248 WEITER LDX XLOW
63C0: A4 01 249 LDY XHIGH
63C2: 86 F9 250 STX TEXL1
63C4: 84 FA 251 STY TEXH1
63C6: A6 06 252 LDX XLOW2
63C8: A4 07 253 LDY XHIGH2
63CA: 86 FB 254 STX TEXL2
63CC: 84 FC 255 STY TEXH2
63CE: A6 1D 256 LDX COUNXL
63D0: A4 1E 257 LDY COUNXH
63D2: 86 FD 258 STX COXL
63D4: 84 FE 259 STY COXH
260 *
261 * Jetzt geht's los!
262 *
63D6: A9 20 263 LOOP LDA #$20
63D8: 85 E6 264 STA HPAG
265 *

```

```

266 * HSCRN-Routine
267 * Ermitteln, ob ein angesprochener
268 * Punkt an oder aus ist ...
269 *
63DA: A6 00 270 LDX XLOW
63DC: A4 01 271 LDY XHIGH
63DE: A5 02 272 LDA Y
63E0: 20 11 F4 273 JSR HPGSN
63E3: B1 26 274 LDA (GBASL),Y
63E5: 25 30 275 AND HMASK
63E7: 29 7F 276 AND #$7F
277 *
278 * ... und dann den entsprechenden
279 * schwarzen oder weißen Punkt auf
280 * die neue Stelle plotten.
281 *
63E9: F0 04 282 BEQ BLACK
63EB: A9 7F 283 LDA #$7F ;weiß
63ED: D0 02 284 BNE POINT
63EF: A9 00 285 BLACK LDA #$0 ;schwarz
63F1: 85 E4 286 POINT STA HCOLORZ
287 *
288 * Grafikseite-Flag (PFLAG) testen
289 * und entsprechende Vorkehrungen
290 * für die Übertragung treffen.
291 *
63F3: A5 FF 292 LDA PFLAG
63F5: F0 04 293 BEQ PAGE1
63F7: A9 40 294 LDA #$40
63F9: 85 E6 295 STA HPAG
63FB: A6 06 296 PAGE1 LDX XLOW2
63FD: A4 07 297 LDY XHIGH2
63FF: A5 08 298 LDA Y2
6401: 20 57 F4 299 JSR HPLOTO
300 *
301 * X-Zähler bis auf 0 dekrementie-
302 * ren, dann nach DOY ...
303 *
6404: C6 1D 304 DEC COUNXL
6406: D0 0E 305 BNE DOX
6408: A5 1E 306 LDA COUNXH
640A: F0 27 307 BEQ DOY
640C: C6 1E 308 DEC COUNXH
309 *
310 * Je nach XFLAG X-Koordinaten
311 * in- oder dekrementieren.
312 *
640E: A5 03 313 DOX LDA XFLAG
6410: D0 0E 314 BNE DECRX
6412: E6 00 315 INC XLOW
6414: D0 02 316 BNE NEXTX
6416: E6 01 317 INC XHIGH
6418: E6 06 318 NEXTX INC XLOW2
641A: D0 BA 319 BNE LOOP
641C: E6 07 320 INC XHIGH2
641E: D0 B6 321 BNE LOOP
322 *
323 * *** TRICK NR. 2 ***
324 *
6420: A5 00 325 DECRX LDA XLOW
6422: D0 02 326 BNE DOL1
6424: C6 01 327 DEC XHIGH
6426: C6 00 328 DOL1 DEC XLOW
6428: A5 06 329 LDA XLOW2
642A: D0 02 330 BNE DOL2
642C: C6 07 331 DEC XHIGH2
642E: C6 06 332 DOL2 DEC XLOW2
6430: 4C D6 63 333 JMP LOOP
334 *
335 * Y-Zähler dekrementieren.
336 * Ist er 0, dann erledigt,
337 * sonst ...
338 *
6433: C6 1F 339 DOY DEC COUNY
6435: F0 2A 340 BEQ RET2
341 *
342 * ... je nach YFLAG Y-Koordinaten
343 * in- oder dekrementieren und ...
344 *
6437: A5 09 345 LDA YFLAG
6439: F0 07 346 BEQ INY
643B: C6 02 347 DEC Y
643D: C6 08 348 DEC Y2
643F: 4C 46 64 349 JMP RSTX
6442: E6 02 350 INY INC Y
6444: E6 08 351 INC Y2
352 *
353 * ... die Start-X-Werte und den
354 * X-Zähler wieder für eine

```

```

355 * neue Runde herstellen.
356 *
6446: A6 F9 357 RSTX LDX TEXTL1
6448: A4 FA 358 LDY TEXH1
644A: 86 00 359 STX XLOW
644C: 84 01 360 STY XHIGH
644E: A6 FB 361 LDX TEXTL2
6450: A4 FC 362 LDY TEXH2
6452: 86 06 363 STX XLOW2
6454: 84 07 364 STY XHIGH2
6456: A6 FD 365 LDX COXL
6458: A4 FE 366 LDY COXH
645A: 86 1D 367 STX COUNXL
645C: 84 1E 368 STY COUNXH
645E: 4C D6 63 369 JMP LOOP
370 *
6461: A9 20 371 RET2 LDA #$20
6463: 85 E6 372 STA HPAG
6465: 60 373 RTS

```

405 Bytes

### HSCRN

```

1 *
2 * HSCRN
3 *
4 *
5 * von N.G. Barbieri/1985
6 *
7 XLOW EQU $00
8 XHIGH EQU $01
9 Y EQU $02
10 FLAG EQU $03
11 GBASL EQU $26
12 HMASK EQU $30
13 HPOSN EQU $F411
14 *
15 * Ermittelt, ob ein Hires-Punkt
16 * weiß oder schwarz ist.
17 * Resultat in FLAG: 0 = schwarz,
18 * nicht 0 = weiß.
19 *
8000: A6 00 20 LDX XLOW
8002: A4 01 21 LDY XHIGH
8004: A5 02 22 LDA Y
8006: 20 11 F4 23 JSR HPOSN
8009: B1 26 24 LDA (GBASL),Y
800B: 25 30 25 AND HMASK
800D: 29 7F 26 AND #$7F
800F: 85 03 27 STA FLAG
8011: 60 28 RTS

```

18 Bytes



**Der nächste Peeker  
Heft 9/1985  
erscheint am  
26. 8. 1985**

## Peeker

Einzelbezug DM 28,-  
Fortsetzungsbezug DM 20,-  
(Jederzeit kündbar, jedoch mindestens  
6 Disketten)

(\* = nur auf Diskette, nicht im Peeker  
gelistet! Seitenangaben beziehen sich  
auf Beginn des Listings)  
Hühig Software Service  
Postfach 102869 · 6900 Heidelberg 1

### Disk # 1

(Heft 1+2, 1984)

T.DISASSEMBLER.65C02 (1/84, S. 15)  
DISASSEMBLER.65C02

T.ACCEL.WAIT (1/84, S. 22)

ACCEL.WAIT  
T.ACCEL.BOOT  
ACCEL.BOOT  
ACCEL.LC.KOPIERER  
T.ACCEL.LC.KOPIE  
ACCEL.LC.KOPIE  
T.ACCEL.ROM.KOPIE1  
ACCEL.ROM.KOPIE1  
T.ACCEL.ROM.KOPIE2  
ACCEL.ROM.KOPIE2

TURTLE.GRAFIK.MIT.REMS (1/84, S.29)  
TURTLE.GRAFIK.OHNE.REMS \*

DOUBLE.LORES.SOFTSWITCH.DEMO  
(1/84, S. 37)  
DOUBLE.LORES.APPLESOFT.DEMO  
AMPER.DOUBLE.LORES.DEMO  
T.AMPER.DOUBLE.LORES  
AMPER.DOUBLE.LORES  
T.DOUBLE.LORES  
DOUBLE.LORES

HIRES (1/84, S. 41)

T.PRINTHIRES  
PRINTHIRES

DHGR.APSOFT.DEMO (2/84, S. 30)  
AMPER.DOUBLE.HIRES.BAS  
AMPER.DOUBLE.HIRES  
T.AMPER.DOUBLE.HIRES  
DHGR.LINEPLOTTER

INSTRING.TEST (2/84, S. 43)

INSTRING.OBJ  
T.INSTRING.OBJ  
INSTRING.LISA.SOURCE

LOESCHEN.EINES.ARRAYS  
(2/84, S. 52)

ULTRATERM.ENGLISCH \* (2/84, S. 60)  
ULTRATERM.DEUTSCH \*

PRIMZAHLEN.OVERMEYER \*  
(2/84, S. 70)  
PRIM.OBJ0 \*  
PRIM.OBJ1 \*

PRIM.TEST \*  
PRIM.TOOLKIT.SOURCE \*

### Disk #2

(Heft 1-2, 1985, DOS-Format)

T.RAMDISKLC (1-2/85, S. 14)  
RAMDISKLC

T.IBS.RAMDISKDRIVER (1-2/85, S. 20)  
IBS.RAMDISKDRIVER

T.AP20.RAMDISKTEST  
AP20.RAMDISKTEST

T.QUICKCOPY (1-2/85, S. 26)  
QUICKCOPY  
QUICKCOPY.PUFFER  
PRODOS.COPYA  
T.PRODOS.COPYOBJ \*  
PRODOS.COPYOBJ

PRODOS.PATCH (1-2/85, S. 31)

T.APPLESOFT.FRE (1-2/85, S. 36)  
T.LC.FRE  
LC.FRE  
FRE.TEST  
T.RAM.FRE \*  
RAM.FRE

T.SCHIRMDISK (1-2/85, S. 44)  
SCHIRMDISK.LISA.SOURCE  
SCHIRMDISK

T.VIDEXT  
VIDEXT.LISA.SOURCE  
VIDEXT

GETPAS (1-2/85, S. 70)

T.GETPAS.ASS \*  
GETPAS.ASS  
GETDOS.PASCAL.SOURCE  
COPYDUPDIR.PASCAL.SOURCE

PRODOS.EDITOR.MACROS  
(1-2/85, S. 86)

### Disk #3

(Heft 1-2, 1985, CP/M-Format)

STEUER.84 (1-2/85, S. 47)  
PASS.BAS  
MENUE.BAS  
HELP.BAS \*

A.BAS  
B.BAS  
C.BAS  
D.BAS  
E.BAS  
F.BAS  
G.BAS  
H.BAS  
I.BAS

# Sammeldisketten

J.BAS K.BAS L.BAS M.BAS N.BAS	SCREEN80.SAVER (4/85, S. 76)	PAGE.SWAP T.PAGE.SWAP	FETT * FETT.INVERSE *
<b>Disk #4</b> (Heft 3 + 4, 1985)	<b>Disk #5</b> (Heft 5, 1985, DOS-Format)	WANDERNDER.STRICH (6/85, S. 16) KOMPRESSOR.DEMO KREIS.1 KREIS.2 KREIS.3 FLIPPER T.FLIPPER KOMPRESSOR T.KOMPRESSOR	PASTOPRO.1D (7/85, S. 62) * PASTOPRO.2D T.PASTOPRO.O PASTOPRO.O
TESTGENERATOR (3/85, S. 26) SAETZE BAHNFAHRT * ZU * TUN.UND.SOLLEN * IRGEND *	T.FM.BSP (5/85, S. 9) FM.BSP	OLYMPIA (6/85, S. 34) T.OLYMPIA	T.CONVERT (7/85, S. 69) CONVERT
MULTIPRECISION (3/85, S. 32)	T.SLOTTRAMDISK (5/85, S. 13) SLOTTRAMDISK SLOTTRAMDISK.HELLO	FOURIER.MAIN (6/85, S. 38) FOURIER.SYN FOURIER.SPEC	T.VORLESER (7/85, S. 71) VORLESER
T.WS.TRANSFER (3/85, S. 36) WS.TRANSFER T.WS.TRANSFER.2 * WS.TRANSFER.2 * GETCPM	PLOT.2.0 (5/85, S. 20) T.PLOT.B PLOT.B PLOT.PROTECTOR	AS.ERWEITERUNG (6/85, S. 43) T.AS.ERWEITERUNG AS.ERW.PROSTART AS.ERW.PRO * T.AS.ERW.PRO *	<b>Disk #8</b> (Heft 8, 1985, DOS-Format)
PRIM.0.SC.SOURCE (3/85, S. 62) PRIM.0.BIN	T.CONVERT560 (5/85, S. 26) CONVERT560 CONVERT560.DEMO	INSTALL.PASCAL.SOURCE (6/85, S. 48) RAMDISK94.PASCAL.SOURCE INIT.PASCAL.SOURCE	HELLO * ASMDIV *
PRIM.1.SC.SOURCE PRIM.1.BIN PRIM.FP	T.EDA (5/85, S. 33) EDA	RAMDISK.INIT.DOS (6/85, S. 55) AUXDRIVER T.AUXDRIVER MOVEDRIVER T.MOVEDRIVER RAMDISK.FORMATTER T.RAMDISK.FORMATTER	DISKTEST (8/85, S. 14) DISKTEST.START T.DISKTEST
ACCELERATOR.ABSTELLEN (3/85, S. 66)	TRANSCEND.PASCAL.SOURCE (5/85, S. 36)	SOLITAIRE.START (6/85, S. 64) SOLITAIRE SOLITAIRE.B T.SOLITAIRE.B	KOPY (8/85, S. 22) BATCHKOPY T.GETSETINFO GETSETINFO BILDTEST
T.WILDCARD.TEST * (3/85, S. 72) WILDCARD.TEST1 * T.WILDCARD.TEST2 * WILDCARD.TEST2 *	T.BLOCKTRACER (5/85, S. 51) BLOCKTRACER T.BLOCKTRACER1 BLOCKTRACER1	RAMDISK.FORMATTER (6/85, S. 55) AUXDRIVER T.AUXDRIVER MOVEDRIVER T.MOVEDRIVER RAMDISK.FORMATTER T.RAMDISK.FORMATTER	T.BOX.COPY (8/85, S. 26) BOX.COPY T.HSCRN HSCRN GRAF.QUATTRO.1
XPLOT.DEMO (4/85, S. 18) XPLOT.ROUTINE T.XPLOT.ROUTINE	FORMAT.LC (5/85, S. 56) FORMAT.LC.START	PYRAMID.PITTY (7/85, S. 6) * T.PYR.PITTY.0 * T.PYR.PITTY.1 * PYR.PITTY.0 * PYR.PITTY.1 * PYR.PITTY.BACK * PYR.PITTY.SHAPE *	T.DOUBLE.LORES (8/85, S. 34) DOUBLE.LORES DOUBLE.LORES.DEMO
MENUE.GENERATOR (4/85, S. 22)	T.DISKDRIVER.DEMO DISKDRIVER.DEMO	T.MEGAWARP.REL (7/85, S. 8) * MEGAWARP.REL * T.MEGAWARP.9900 MEGAWARP.9900 T.SPEEDTEST SPEEDTEST	START.CMD * (8/85, S. 40) HMENUE.CMD * AUFNAHME.CMD * AUFMASKE.CMD * AUSGABE.CMD * SUCH.CMD * EDITFNAME.CMD * SUCHVNAME.CMD * SUCHBEME.CMD * SCHREIBA.CMD * SCHREIBL.CMD * LOESCH.CMD *
T.MACROS.65C02 (4/85, S. 31)	RANDOM.DEMO (5/85, S. 69) COLUMN80.DEMO	FORMAT (7/85, S. 20) T.FORMAT.OBJ FORMAT.OBJ	IDSEARCH.PASCAL.SOURCE (8/85, S. 49)
TERMINAL (4/85, S.36) TERMINAL.B T.TERMINAL.B	SUPERDUMP.EPSON (6/85, S. 22!) SUPERDUMP.IMAGEWRITER SUPERDUMP.BILD T.SUPERDUMP * SUPERDUMP EPSON IMAGEWRITER	BITEDITOR (7/85, S. 29) NORMAL *	FAKULTAET.DEMO (8/85, S. 57) T.FAKULTAET FAKULTAET
CAT.ARRAY (4/85, S. 44) CAT.SAVER EINTRAG.SUCHER EINTRAG.ANALYSE PRODOS.READER T.PRODOS.READER.OBJ PRODOS.READER.OBJ	<b>Disk #6</b> HELLO (6/85, S. 72) * ASMDIV *		GRAFIK.DEMOS (8/85, S. 68) ZEICHENJAGD (8/85, S. 70)
MOUSESTUFF.PASCAL.SOURCE (4/85, S. 51) MOUSE.ASS.PASCAL.SOURCE TESTMOUSE.PASCAL.SOURCE DRAWMOUSE.PASCAL.SOURCE	CURSOR1 (6/85, S. 6) T.CURSOR1 CURSOR2 T.CURSOR2 LINIE T.LINIE VIERECK T.VIERECK BOX T.BOX HINTERGRUND T.HINTERGRUND		T.RAM.FRE.NEU (8/85, S. 70) * RAM.FRE.NEU
INALL.DATA (4/85, S. 70) SCREEN80.DATA (4/85, S. 33)			



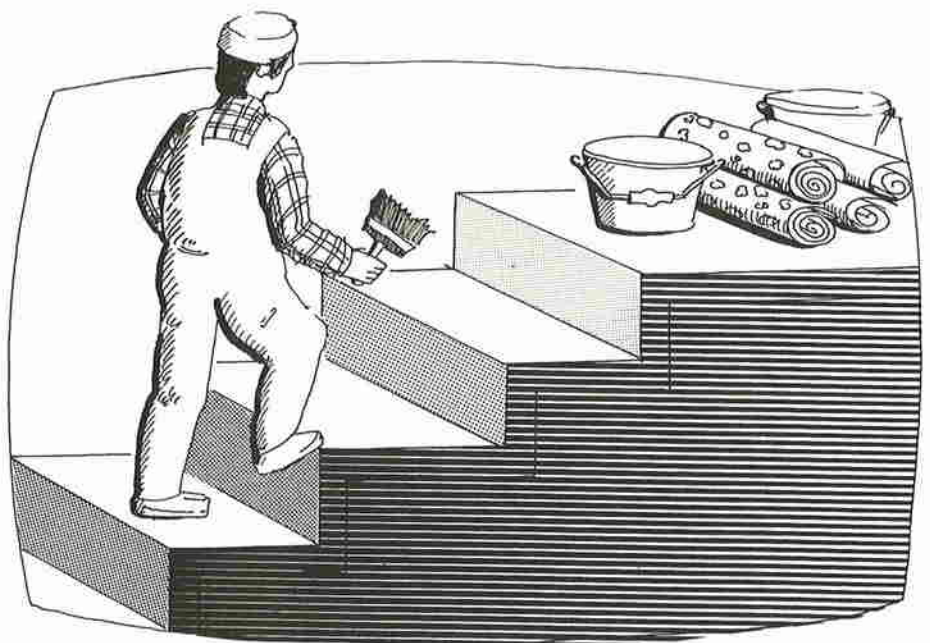
# Double-Lores- Applesoft- Erweiterungen

von Karl-Walter Bott

Wenn Sie einen Apple IIe mit einer erweiterten 80-Zeichenkarte oder einen Apple IIc besitzen, dann können Sie mit der hier vorgestellten Software den Applesoft-Interpreter um sechs neue Grafik-Befehle erweitern.

Die erweiterte 80-Zeichenkarte für den Apple IIe stellt zu den bereits vorhandenen 64K RAM auf dem Motherboard weitere 64K RAM zur Verfügung, die sich auf vielfältige Weise einsetzen lassen. Eine dieser Möglichkeiten besteht darin, die Auflösung der Lores-Grafik zu verdoppeln, so daß 80 mal 48 Kästchen darstellbar sind. Über die Organisation und die Programmierung der Double-Lores-Grafik wurde bereits im *Peeker 1/84* ausführlich berichtet.

Die Programmierung von Double-Lores in Applesoft-BASIC erweist sich als sehr umständlich, da der Interpreter keine Befehle zur Erstellung von Double-Lores-

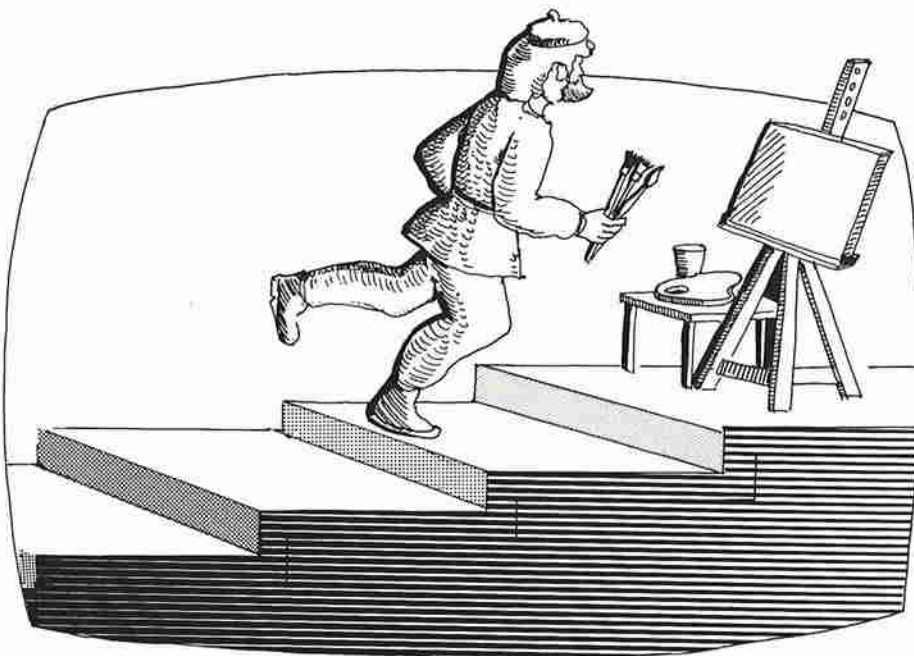


Grafiken enthält, wie man sie von der normalen niedrigauflösenden Grafik gewohnt ist.

Das hier vorgestellte Software-Paket implementiert die fehlenden Befehle, so daß Applesoft-Programmierer die Double-Lores-Grafik nutzen können, ohne detaillierte Kenntnisse über deren rechnerinterne Organisation zu besitzen.

schaltet und die erweiterte 80-Zeichenkarte deaktiviert.

Dieser Befehl sollte nur dann benutzt werden, wenn die 80-Zeichenkarte vor Aufruf von &GR nicht aktiv war, da ansonsten Softswitch-Probleme auftreten. Wurde die Double-Lores vom 80-Zeichenmodus aus aktiviert, genügt ein TEXT-Befehl zum Zurückschalten.



## Die neuen Befehle

**&GR** – Das &GR-Statement veranlaßt den Computer, auf doppelt-niedrigauflösende Grafik umzuschalten. Dabei wird der Bildschirm gelöscht und vier Textzeilen am unteren Bildschirmrand bereitgestellt. Die Darstellung auf dem Bildschirm erfolgt jetzt in einem 80-mal-40-Raster.

Will man die ganze Seite mit Grafik ausfüllen (80 mal 48 Punkte), so muß zunächst durch

POKE 49234,0

auf Full-Screen-Ausgabe umgeschaltet werden. Um die acht Grafikzeilen am unteren Bildschirmrand zu löschen, verwende man dann folgende Befehle:

POKE 49236,0: CALL 63538

POKE 49237,0: CALL 63538.

Hierdurch wird der ganze Bildschirm gelöscht und wie oben die Farbe schwarz eingestellt (COLOR = 0).

**&TEXT** – Dieser Befehl schaltet von Double-Lores-Grafik auf Textdarstellung um. Es wird der 40-Zeichen-Textmodus einge-

**&PLOT** – Der &PLOT-Befehl zeichnet ein Kästchen in der aktuellen Farbe auf dem Double-Lores-Bildschirm. Der erste Ausdruck, der dem Schlüsselwort „&PLOT“ folgt, gibt die Spalte an, in der das Kästchen gezeichnet werden soll (zwischen 0...79, von links nach rechts); der zweite Ausdruck, durch ein Komma vom ersten getrennt, bestimmt die Zeile (0...47, von oben nach unten).

Beispiel: „&PLOT 10, 15“ plottet einen Block in Spalte 10, Zeile 15.

Der normale PLOT-Befehl wurde dahingehend erweitert, daß jetzt – wie unter HGR – auch Linien zwischen zwei oder mehreren beliebigen Punkten gezeichnet werden können.

Beispiel: „&PLOT 0, 0 TO 79, 47“ zeichnet eine diagonale Linie von links oben nach rechts unten in der aktuellen Farbe. Liegen die angegebenen Koordinaten außerhalb des zulässigen Bereichs, wird das Programm unterbrochen und die Meldung „ILLEGAL QUANTITY ERROR“ ausgegeben.

**&HLIN** – Das &HLIN-Statement zeichnet eine horizontale Linie in der aktuellen Farbe. Die beiden Ausdrücke, die dem Befehl „&HLIN“ folgen, bestimmen die Spaltennummern, in denen der Anfang und das Ende der Linie liegen. Der Ausdruck, der dem Schlüsselwort „AT“ folgt, bestimmt die Zeile, in welcher die horizontale Linie gezeichnet werden soll.

Beispiel: „&HLIN 20, 60 AT 10“ zeichnet eine horizontale Linie in Zeile 10 von Spalte 20 bis Spalte 60.

Wird die zulässige Spalten- oder Zeilenzahl überschritten, wird eine Fehlermeldung ausgegeben und das Programm unterbrochen.

**&VLIN** – Das &VLIN-Statement zeichnet eine vertikale Linie. Die beiden folgenden Ausdrücke bestimmen Anfangs- und Endzeile der Linie, während der Ausdruck nach „AT“ die Spalte bestimmt, in der die Linie gezeichnet wird.

Beispiel: „&VLIN 10, 30 AT 20“ zeichnet eine vertikale Linie in Spalte 20 von Zeile 10 bis Zeile 30.

Bei Überschreiten der zulässigen Spalten- oder Zeilenzahl wird eine Fehlermeldung ausgegeben und der Programmablauf unterbrochen.

**&SCRN** – Der &SCRN-Befehl ermittelt die Farbe eines angegebenen Blocks auf dem Double-Lores-Bildschirm. Der Wert, den die Funktion zurückgibt, liegt zwischen 0 und 15, und wird der angegebenen Variablen zugewiesen, die wahlweise vom Typ Integer oder Real sein kann.

Beispiel: „&SCRN(A, 50, 10)“ ermittelt die Farbe des Blocks in Spalte 50, Zeile 10. Die Variable A hat dann einen Wert zwischen 0 und 15.

Wenn die Variable nicht dem zulässigen Typ entspricht (z.B. A\$) oder die zulässigen Grenzen für die Koordinatenangaben überschritten werden, erfolgt eine Fehlermeldung.

Der normale COLOR-Befehl kann weiterhin verwendet werden, um die gewünschte Farbe eines Blockes zu definieren, wobei der Ausdruck rechts vom Gleichheitszeichen zwischen 0 und 15 liegen muß.

Beispiel: „COLOR = 15“ wählt die Farbe weiß aus.

## Technische Anmerkungen

Das Programm residiert ab Adresse \$931A im Hauptspeicher und ist 742 Bytes

lang, kann aber durchaus auch mit einer anderen Startadresse assembliert werden. (Anm.: Dies ist insbesondere für ProDOS erforderlich. Unter DOS 3.3 liegt das Programm unterhalb von \$9600, d.h. im Bereich \$931A-\$95FF unterhalb vom dritten DOS-Puffer bei Maxfiles 3. Man *könnte* es auch unterhalb von \$9A00 legen und die DOS-Puffer entsprechend nach unten verschieben. Unter ProDOS *muß* es unterhalb von \$9A00 liegen, d.h. im Bereich \$971A-\$99FF, weil es sonst durch einen Textfile-Zugriff zerstört würde. Zu diesem Zweck muß man über die GETBUFR-Routine 3 Pages anfordern und dann das Programm mit BRUN DOUBLE.LORES starten. Aus all diesen Gründen wird HIMEM durch DOUBLE.LORES nicht geändert. us)

Um die neuen Applesoft-Befehle in den Interpreter einzubinden, wurde der Ampersand-Vektor verwendet. Er bietet eine bequeme Möglichkeit, eigene Routinen ohne Geschwindigkeitsverlust an den Interpreter anzuhängen.

Jedesmal, wenn der Interpreter auf das &-Zeichen trifft, verzweigt er zur Adresse \$03F5, in der normalerweise ein Sprung zu einem RTS-Befehl steht. An dieser Stelle wird nun ein Sprung zum eigenen Programm eingetragen.

Stößt der Interpreter auf das &-Zeichen, so verzweigt er zum Anwenderprogramm, das zunächst überprüft, welcher Token vorliegt, um zu dem entsprechenden Programmsegment zu springen. Nach Ausführung des jeweiligen Befehls wird der Textpointer bis zum nächsten Applesoft-Befehl vorgerückt und DOUBLE.LORES über einen RTS-Befehl verlassen.

### Konflikte mit 40/80-Z/Z

Es empfiehlt sich, vor Aktivierung der Double-Lores die 80-Zeichenkarte mit „PRINT CHR\$(21)“ oder „ESC Ctrl-Q“ abzustellen, um Speicherkonflikte zu vermeiden und das korrekte Arbeiten des &TEXT-Befehls zu gewährleisten.

Sollten Sie keine Double-Lores-Grafik auf dem Bildschirm sehen, kann das an Ihrer erweiterten 80-Zeichenkarte liegen. Entweder Sie besitzen eine Nachbau-80-Zeichenkarte, die nicht über die entsprechenden technischen Möglichkeiten verfügt, oder die Karte ist nicht korrekt installiert. Lesen Sie dazu das entsprechende Kapitel im Handbuch zur 80-Zeichenkarte.

Bei gemischter Darstellung (Text und Grafik) beachten Sie bitte folgendes:

### DOUBLE.LORES

```

1          ORG $931A
2          *
3          *   Double-Lores  1.0
4          *
5          *
6          *   K.-W. Bott, 1985
7          *
8          *
9          *   Tokens
10         *
11         TFLOT EQU $8D
12         TGR   EQU $88
13         THLIN EQU $8E
14         TVLIN EQU $8F
15         TSCRN EQU $D7
16         TTEXT EQU $89
17         TO    EQU $C1
18         AT    EQU $C5
19         *
20         *   Applesoft-Routinen
21         *
22         AMPER EQU $03F5 ;Ampersand-Vektor
23         TABV  EQU $FB5B ;Cursor vertikal tabulieren
24         CLRTOP EQU $FB36 ;Lores-Schirm löschen
25         PLOT  EQU $FB00 ;Block plotten
26         SCRNL EQU $FB71 ;Lores-Screen-Befehl
27         CHRGET EQU $00B1 ;Zeichen aus BASIC
28         GETBYT EQU $E6F8 ;Byte aus BASIC
29         SYNCHR EQU $DECO ;Zeichen im AREG
30         * ;mit BASIC vergleichen
31         SNERR  EQU $DEC9 ;"SYNTAX ERROR"
32         ILQUAN EQU $E199 ;"ILLEGAL QUANTITY"
33         PTRGET EQU $DFE3 ;Pointer auf Variable
34         * ;aus BASIC-Text
35         SNGFLT EQU $E301 ;YREG -> FAC
36         *
37         *   Konstanten
38         *
39         PAGE1 EQU $C054 ;Hauptspeicher
40         PAGE2 EQU $C055 ;Zusatzspeicher
41         CHAR  EQU $B8 ;aktuelles Zeichen im
42         * ;BASIC-Text
43         POINTER EQU $CE ;Zeiger für Puffer
44         VARNAM EQU $81 ;letzte BASIC-Variable
45         SUBFLG EQU $14 ;Flag für Var.-behandlung
46         FAC    EQU $9D ;Float.-Akkumulator
47         WNDTOP EQU $22 ;obere Begrenzung des
48         * ;Scroll-Windows
49         *
50         *   Speicherplatz für Koordinaten
51         *
52         X1    EQU $06
53         X2    EQU $07
54         Y1    EQU $09
55         Y2    EQU $E3
56         XS    EQU $D7 ;Schrittweite X
57         YS    EQU $EB ;Schrittweite Y
58         XD    EQU $CF ;Differenz X1 - X2
59         YD    EQU $EF ;Differenz Y1 - Y2
60         *
61         *
62         *   Ampersand-Vektor linken
63         *
931A: A9 4C 64 START LDA #$4C ;JMP
931C: 8D F5 03 65 STA AMPER
931F: A9 2A 66 LDA #<LINK
9321: 8D F6 03 67 STA AMPER+1
9324: A9 93 68 LDA #>LINK
9326: 8D F7 03 69 STA AMPER+2
9329: 60 70 RTS
71         *
72         *   Einsprung für jeden Ampersand-
73         *   Befehl; Token im A-Register
74         *
932A: C9 88 75 LINK CMP #TGR
932C: F0 20 76 BEQ GR
932E: C9 89 77 CMP #TTEXT
9330: F0 46 78 BEQ TEXT
9332: C9 8D 79 CMP #TFLOT
9334: F0 0F 80 BEQ JPLOT
9336: C9 8E 81 CMP #THLIN
9338: F0 54 82 BEQ HLIN
933A: C9 8F 83 CMP #TVLIN
933C: F0 0A 84 BEQ JVLIN
933E: C9 D7 85 CMP #TSCRN
9340: F0 09 86 BEQ JSCRN

```



1. Ein &PLOT-Befehl über die 39. Grafikkarte hinaus druckt ein Zeichen in die unteren vier Textzeilen.
2. Die erweiterte 80-Zeichenkarte sollte vor Aktivierung der Double-Lores-Grafik eingeschaltet sein und eingeschaltet bleiben.
3. War die 80-Zeichenkarte aktiviert, so ist der TEXT-Befehl zum Zurückschalten zu verwenden, ansonsten &TEXT.

Bei reiner Grafikdarstellung ist folgendes zu berücksichtigen:

1. Benutzen Sie die oben erwähnten POKE- und CALL-Befehle, um die unteren vier Zeilen für Grafik freizuhalten.
2. Jeder PRINT-Befehl druckt Zeichen in den Bildschirmspeicher, die als Farbblöcke auftreten.
3. Selbst ein GET-Befehl erzeugt einen farbigen Block, da der Cursor sichtbar wird. Um dies zu vermeiden, kann die Eingabe eines Zeichens von der Tastatur z.B. wie folgt vorgenommen werden:  
10 ON PEEK (49152) < 128 GOTO 10:A\$ = CHR\$(PEEK (49152) - 128): POKE 49168,0
4. Für das Zurückschalten gilt das gleiche wie oben.

Das Programm **DOUBLE.LORES.DEMO** zeigt eine Anwendung der neuen Befehle und die Umschaltung auf reine Grafik-Ausgabe.



```

9342: 4C C9 DE 87      JMP SNERR
          *
9345: 4C 1B 94 89      JPL0T  JMP SPLOT
9348: 4C C0 93 90      JVLIN  JMP VLIN
934B: 4C F1 93 91      JSCRN  JMP SCREEN
          *
          * &GR
          * ==
          *
          * Double-Lores initialisieren
          * und Bildschirm löschen
          *
934E: AD 50 C0 99      GR      LDA $C050 ;GRAFIK
9351: AD 56 C0 100     LDA $C056 ;LORES
9354: AD 53 C0 101     LDA $C053 ;MIXSET
9357: 8D 01 C0 102     STA $C001 ;80STORE
935A: 8D 0D C0 103     STA $C00D ;80COL
935D: AD 5E C0 104     LDA $C05E ;AN3
9360: A9 14 105       LDA #20
9362: 85 22 106       STA WNDTOP ;Scroll-Window begrenzen
9364: A9 17 107       LDA #23 ;Cursor in Zeile 23
9366: 20 5B FB 108     JSR TABV ;setzen
9369: AD 55 C0 109     LDA PAGE2
936C: 20 36 FB 110     JSR CLRTOP ;Zusatzspeicher löschen
936F: AD 54 C0 111     LDA PAGE1
9372: 20 36 FB 112     JSR CLRTOP ;Hauptspeicher löschen
9375: 4C B1 00 113     JMP CHRGET ;Ttxtptr. vorrücken
          *
          *
          * &TEXT
          * ==
          *
          * Zurück in den Textmodus
          *
9378: 8D 00 C0 121     TEXT   STA $C000 ;40STORE
937B: 8D 0C C0 122     STA $C00C ;40COL
937E: AD 5F C0 123     LDA $C05F ;AN3 OFF
9381: AD 51 C0 124     LDA $C051 ;TEXT
9384: AD 54 C0 125     LDA PAGE1
9387: A9 00 126       LDA #0
9389: 85 22 127       STA WNDTOP ;volles Scroll-Window
938B: 4C B1 00 128     JMP CHRGET ;Ttxtptr. ink. -> BASIC
          *
          * &HLIN X1, X2 AT Y
          * ==
          *
938E: 20 B1 00 133     HLIN   JSR CHRGET ;TXTPTR vorrücken
9391: 20 FB E6 134     JSR GETBYT ;Param. X1 aus BASIC-Text
9394: E0 50 135       CPX #80 ;X1 > 79?
9396: B0 25 136       BCS ERROR
  
```

## APPLE-INTERFACE

Weitgehend deutsche Produktion

MESON Ii-PC, 48K, Apple-kompatibel ab 1000,-  
Z80, 16K je 85,-, 95,-  
PAL, EPROM, CENTRONICS, RS 232 je 170,-  
80 Zeichen SoftSwitch 200,-  
ADD 2B, VIA-Card (6522) mit RAM u. Backup 170,-  
ADD 4B, FIA-Card (6821) mit RAM u. Backup 150,-  
JOYSTICK zum Anschluss von zwei Aaht-  
Joystick 40,-  
TEAC FD 55 F 550,-  
Shugart SA 465 80 Track OS sehr leise 300,-  
F-Print-Controller APCD4 Autopatch 300,-  
LDD 103, 40 Track Apple-kompatibel Slim 420,-  
Pette pro Stick, inkl. Mvst.  
Ladegerät, Mo. 12-18 Uhr, Di.-Do. 15-18 Uhr,  
Sa. 10-13 Uhr  
Telefon 030-883 1303 (Vire Ladegerät)

## KLEINTEILE

Stand 29. 5. 1985

UND 4116-200ns 1-9  
TMR 4128-150ns 1-30  
TMM 41286 C15 1-30  
6116 LP3 (SRM 2016) 1-30  
2704-490ns 1-30  
2725-490ns 1-30  
2761-250ns 1-14  
27126-250ns 1-14  
27286-250ns 1-14  
Z41500 1-19  
Z41500 1-19  
TDA2030 1-19  
74HC00 1-19  
LED Strm Rot superhell 10 Stick 1-19  
LED 5mm Rot blink 1-19  
LED 5mm Grün o. Gelb blink 1-19  
Viele Bausteine und Mikrocontroller a. A. Mindestbestellmenge 50,-  
DM, sonst +10,- DM, Porto und Verpackung bis 250 g 5,- DM  
Versand per NV oder VK, bei NV + 1,50 DM

T. Tank & G. Köhler · Gedröschulzenweg 72 · 1000 Berlin 45

# TMBSTONE-MICRO

## Apple und IBM kompatible Computer

16K, Z80, Diskcontroller je .....	110,-
80 Zeichenkarte mit Softswitch	
2 Zeichensätze .....	195,-
Motherboard 48K ohne	
Firmware .....	610,-
Erphi-controller mit Autopatch .....	300,-
Siemenslaufwerk F 122 .....	515,-
Philips X3134 2x80 Track .....	465,-
TEAC FD-55B 2x40 Track .....	468,-
TEAC FD-55F 2x80 Track .....	565,-

**Neu: Stardrucker SG 10 .....** **920,-**

Monochrome Monitore .....

Farbmonitore .....

Tastaturen für IBM und Apple .....

Versand nur per Nachnahme oder Vorkasse

Weiteres Zubehör für Apple und IBM gegen frankierten Rückumschlag.

**Preisenkung:**  
**128K Karte (Saturn kompatibel) .... 395,-**

Zusatzkarten und Motherboard ausnahmslos deutsche Fertigung mit ausgesuchten Bauteilen.

**Ulf Mohwinkel Electronic**  
Berliner Straße 73  
5090 Leverkusen Fettehenne  
Telefon 02 14/9 37 81

# APPLE //e

# APPLE IIc

und kompatible Computer

Universeller EPROM-Programmer 4003

- Programmiert alle gängigen EPROM-Typen (z.B.: 2716, 2732, -64, -128, 2508, -16, -32, -64...)
- Voll menügesteuerte Software auf Diskette
- Kein Schalten, Löten oder Stecken nötig
- Programmierspannung wird im Gerät erzeugt
- Verbindung zum Rechner über Flachbandkabel
- Rote und grüne Leuchtdiode zur Betriebsart-Anzeige
- Komplett mit 28 poligem Textool-Socket

Fertigerät DM 269,50 ■ Bausat + Anleitung DM 219,-

## APPLE //e

## + 64 KByte RAM

- 80 gestochen scharfe und flimmerfreie Zeichen / Zeile
- plus zusätzliche 64 KByte schnelle dynamische RAM's
- ermöglicht Double Hires Graphik ( 560 \* 192 Punkte )
- vergoldete Steckerleiste und Qualitätsbauteile

Geprüfte Platine + Demo Disk + Beschreibung DM 174,50  
Bausatz DM 145,- ■ Leerplatine + Bauleitung DM 59,-  
Ab Lager lieferbar. Alle Preise inklusive Mehrwertsteuer.

## DOBBERTIN

### INDUSTRIE-ELEKTRONIK

Brahmsstraße 9, 6835 Brühl, Tel. (06202) 71417

```

9398: 86 06 STX X1
939A: A9 2C LDA #', '
939C: 20 C0 DE JSR SYNCHR
939F: 20 F8 E6 JSR GETBYT
93A2: E0 50 140 CFX #80
93A4: 86 07 STX X2
93A6: A9 C5 LDA #AT
93A8: 20 C0 DE JSR SYNCHR
93AB: 20 F8 E6 JSR GETBYT
93AE: E0 30 146 CFX #48
93B0: B0 0B BCS ERROR
93B2: 86 09 STX Y1
93B4: 86 E3 STX Y2
93B6: 20 8A 94 JSR LINPLOT
93B9: AD 54 C0 LDA PAGE1
93BC: 60 152 RTS
93BD: 4C 99 E1 JMP ILQUAN
154 *
155 * &VLIN Y1, Y2 AT X
156 *
157 *
VLIN JSR CHRGET
93C3: 20 F8 E5 JSR GETBYT
93C6: E0 30 160 CFX #48
93C8: B0 F3 BCS ERROR
93CA: 86 09 STX Y1
93CC: A9 2C LDA #', '
93CE: 20 C0 DE JSR SYNCHR
93D1: 20 F8 E6 JSR GETBYT
93D4: E0 30 166 CFX #48
93D6: B0 E5 BCS ERROR
93D8: 86 E3 STX Y2
93DA: A9 C5 LDA #AT
93DC: 20 C0 DE JSR SYNCHR
93DF: 20 F8 E6 JSR GETBYT
93E2: E0 50 172 CFX #80
93E4: B0 D7 BCS ERROR
93E6: 86 06 STX X1
93E8: 86 07 STX X2
93EA: 20 8A 94 JSR LINPLOT
93ED: AD 54 C0 LDA PAGE1
93F0: 60 178 RTS
179 *
180 * &SCRN (VAR, X, Y)
181 *
182 *
183 * Farbwert des Blockes X, Y bestimmen
184 * und in BASIC-Variablen übertragen
185 *
SCREEN JSR CHRGET
93F4: 20 AD 95 JSR GETADR
188 *
LDA #', '
93F7: A9 2C LDA #', '
93F9: 20 C0 DE JSR SYNCHR
93FC: 20 58 94 JSR GETVAL
93FF: A5 06 LDA X1
9401: 4A 193 LSR
9402: 90 05 BCC S2
9404: AE 54 C0 LDA PAGE1
9407: E0 03 BCS OKAY
9409: AE 55 C0 LDX PAGE2
940C: A8 198 TAY
940D: A5 09 LDA Y1
940F: 20 71 F8 JSR SCRN
9412: 20 D2 95 JSR GIVEBACK

```

```

202 *
203 LDA PAGE1
204 JMP CHRGET
205 *
206 *
207 *
208 * &PLOT X1, Y1 TO X2, Y2 ....
209 *
210 *
SPLIT JSR CHRGET
941B: 20 B1 00 LDX #0
941E: A2 00 LDA (CHAR,X)
9420: A1 B8 LDA #TO
9422: C9 C1 CMP #TO
9424: F0 0E BEQ CTO
9426: 20 58 94 JSR GETVAL
9429: 20 73 94 JSR EXPLOT
218 *
219 PLOTTO LDX #0
942C: A2 00 LDA (CHAR,X)
942E: A1 B8 LDA #TO
9430: C9 C1 CMP #TO
9432: D0 20 BNE EXIT
223 *
224 CTO JSR CHRGET
9434: 20 B1 00 JSR CHRGET
225 *
226 * Neue Param. holen und Xn, Yn <-> Xn+1, Yn+1
227 *
228 LDA X1
9437: A5 06 PHA
229 LDA Y1
9439: 48 PHA
230 LDA Y1
943A: A5 09 PHA
231 PHA
943C: 48 JSR GETVAL
232 LDA X1
943D: 20 58 94 JSR GETVAL
233 LDA X2
9440: A5 06 STA X2
234 LDA Y1
9442: 85 07 STA X2
235 LDA Y2
9444: A5 09 STA Y2
236 PLA
9446: 85 E3 STA Y1
237 PLA
9448: 68 STA Y1
238 PLA
9449: 85 09 STA X1
239 PLA
944B: 68 STA X1
240 STA X1
241 *
944E: 20 8A 94 JSR LINPLOT
9451: 4C 2C 94 JMP PLOTTO
243 *
244 *
245 EXIT LDA PAGE1
9454: AD 54 C0 RTS
246 *
247 *
248 * GETVAL holt die X- und Y-Parameter
249 * aus dem BASIC-Text
250 *
GETVAL JSR GETBYT
9458: 20 F8 E6 CFX #80
945B: E0 50 BCS FEHLER
945D: B0 11 STX X1
253 STX X1
945F: 66 06 LDA #', '
254 LDA #', '
9461: A9 2C JSR SYNCHR
9463: 20 C0 DE JSR GETBYT
9466: 20 F8 E6 JSR GETBYT
9469: E0 30 CFX #48
258 BCS FEHLER
946B: E0 03 STX Y1
946D: 86 09 RTS
261 *
262 FEHLER JMP ILQUAN
9470: 4C 99 E1
263 *
264 * EXPLOT plottet einen Punkt mit
265 * den Koordinaten X1, Y1 auf dem
266 *

```



```

94D7: F0 I7      332      BEQ NOCOR2      ;ETERM = 0
94D9: 18         333      CLC
94DA: A5 06     334      LDA X1
94DC: 65 D7     335      ADC XS
94DE: 85 06     336      STA X1
94E0: 38       337      SEC
94E1: A5 CF     338      LDA XD
94E3: E5 EF     339      SBC YD
94E5: 0A       340      ASL
94E6: 18       341      CLC
94E7: 6D A9 95 342      ADC ETERM
94EA: 8D A9 95 343      STA ETERM
94ED: 4C FA 94 344      JMP LAB3
94F0: A5 CF     345      LDA XD
94F2: 0A       346      ASL
94F3: 18       347      CLC
94F4: 6D A9 95 348      ADC ETERM
94F7: 8D A9 95 349      STA ETERM
94FA: 18       350      LAB3
94FB: A5 EB     351      LDA YS
94FD: 65 09     352      ADC Y1
94FF: 85 09     353      STA Y1
9501: 4C CC 94 354      JMP LOOP2
9504: 4C 73 94 355      JMP EXPLOT
          356      *
          357      * BRENSHAM-Algorithmus XD >= YD
          358      * hier ETERM eine 16-Bit-Signed-Number
          359      *
9507: A5 EF     360      LDA YD
9509: 8D AB 95 361      STA TMP
950C: A9 00     362      LDA #0
950E: 8D AC 95 363      STA TMP+1
9511: 0E AB 95 364      ASL TMP
9514: 2E AC 95 365      ROL TMP+1
9517: 38       366      SEC
9518: AD AB 95 367      LDA TMP
951B: E5 CF     368      SBC XD
951D: 8D A9 95 369      STA ETERM
9520: A9 00     370      LDA #0
9524: 8D AA 95 372      SBC #0
9527: A6 CF     373      STA ETERM+1
9529: E8       374      LDY XD
952A: CA       375      INX
          376      LOOP
952B: F0 5D     377      DEX
952D: 20 73 94 378      BEQ EXPLOT
9530: AD AA 95 379      LDA ETERM+1
9533: 30 39     380      BMI NOCOR
9535: D0 05     381      BNE MARK
9537: AD A9 95 381      LDA ETERM
953A: F0 32     382      BEQ NOCOR
953C: 18       383      CLC
953D: A5 09     384      LDA Y1
953F: 65 EB     385      ADC YS
9541: 85 09     386      STA Y1
9543: 38       387      SEC
9544: A5 EF     388      LDA YD
9546: E5 CF     389      SBC XD
9548: 8D AB 95 390      STA TMP
954B: A9 00     391      LDA #0
954D: E9 00     392      SBC #0
954F: 8D AC 95 393      STA TMP+1
9552: 0E AB 95 394      ASL TMP
9555: 2E AC 95 395      ROL TMP+1
          396      CLC

```

```

267 * Double-Lores-Schirm
268 *
269 EXPLOT LDA X1
270 LSR
271 BCC PAGE2
272 LDY PAGE1
273 BCS CONT
274 LDY PAGE2
275 CONT
276 TAY
277 LDA Y1
278 JSR PLOT
279 LDA PAGE1
280 RTS
281 * LINPLOT plottet eine Linie
282 * zwischen zwei Punkten
283 *
284 LINPLOT LDA #1
285 STA XS
286 STA YS
287 *
288 SEC
289 LDA X2
290 SBC X1
291 STA XD
292 *
293 SEC
294 LDA Y2
295 SBC Y1
296 STA YD
297 *
298 * Setzen der Schrittweite für Liniengenerator
299 *
300 BIT XD
301 BPL LAB1
302 BIT YD
303 BMI CASE1
304 BPL CASE2
305 BIT YD
306 BMI CASE3
307 BPL FERTIG
308 JSR MIYS
309 CASE2 JSR MIXS
310 JMP FERTIG
311 CASE3 JSR MIYS
312 *
313 FERTIG LDA XD
314 CMP YD
315 BPL DDA
316 *
317 * BRENSHAM-Algorithmus für YD > XD
318 * (Modifizierter Digital-Differential-Analyzer DDA)
319 *
320 LDA XD
321 ASL
322 SEC
323 SEC YD
324 STA ETERM
325 LDY YD
326 INX
327 LOOP2 DEX
328 BEQ READY2
329 JSR EXPLOT
330 LDA ETERM
331 BMI NOCOR2

```

```

9473: A5 06
9475: 4A
9476: 90 05
9478: AC 54 C0
947B: 50 03
947D: AC 55 C0
9480: A8
9481: A5 09
9483: 20 00 F8
9486: AD 54 C0
9489: 60
948A: A9 01
948C: 85 D7
948E: 85 EB
9490: 38
9491: A5 07
9493: E5 06
9495: 85 CF
9497: 38
9498: A5 E3
949A: E5 09
949C: 85 EF
949E: 24 CF
94A0: 10 06
94A2: 24 EF
94A4: 30 08
94A6: 10 09
94A8: 24 EF
94AA: 30 0B
94AC: 10 0C
94AE: 20 8D 95
94B1: 20 9B 95
94B4: 4C BA 94
94B7: 20 8D 95
94BA: A5 CF
94BC: C5 EF
94BE: 10 47
94C0: A5 CF
94C2: 0A
94C3: 38
94C4: E5 EF
94C6: 8D A9 95
94C9: A6 EF
94CB: E8
94CC: CA
94CD: F0 35
94CF: 20 73 94
94D2: AD A9 95
94D5: 30 19

```



```

9559: AD AB 95 397 LDA TMP
9560: 8D A9 95 398 ADC ETERM
9561: 6D A9 95 399 STA ETERM
9562: AD AC 95 400 LDA TMP+1
9563: 6D AA 95 401 ADC ETERM+1
9564: 8D AA 95 402 STA ETERM+1
9565: 4C 80 95 403 JMP LAB2
9566: A5 EF 404 NOCOR LDA YD
9567: 0A 405 ASL
9568: 18 406 CLC
9569: 6D A9 95 407 ADC ETERM
9570: 8D A9 95 408 STA ETERM
9571: AD AA 95 409 LDA ETERM+1
9572: 69 00 410 ADC #0
9573: 8D AA 95 411 STA ETERM+1
9574: 18 412 LAB2 CLC
9575: A5 D7 413 LDA XS
9576: 65 06 414 ADC X1
9577: 85 06 415 STA X1
9578: 4C 2A 95 416 JMP LOOP
9579: 4C 73 94 417 READY1 JMP EXPLOT
9580: 418 * MIYS LDA #FFF
9581: A9 FF 419 STA YS
9582: 85 EB 420 ;YS = -1
9583: A5 EF 421 LDA YD
9584: 49 FF 422 EOR #%11111111
9585: 18 423 CLC
9586: 69 01 424 ADC #1
9587: 85 EF 425 STA YD
9588: 85 EF 426 RTS
9589: 60 427 * MLXS LDA #FFF
9590: A9 FF 428 STA XS
9591: 85 D7 429 ;XS = -1
9592: A5 CF 430 LDA XD
9593: 49 FF 431 EOR #%11111111
9594: 18 432 CLC
9595: 69 01 433 ADC #1
9596: 85 CF 434 STA XD
9597: 85 CF 435 RTS
9598: 60 436 ETERM HEX 00
9599: 00 437 ;Error-Term für DDA
9600: 00 438 TMP HEX 00
9601: 00 439 ;Hilfsspeicher für 16-Bit-Mult.
9602: 00 440 *
9603: 441 * Adresse für Variable, die den Farbcode
9604: 442 * erhalten wird, feststellen und
9605: 443 * Variablentyp (Real, Integer) bestimmen
9606: 444 *
9607: A9 00 445 GETADR LDA #0
9608: 85 14 446 STA SURFLG
9609: 20 E3 DF 447 JSR PTRGET
9610: 448 *
9611: 85 CE 449 STA POINTER+1
9612: 84 CF 450 STY POINTER+1
9613: 451 *
9614: A5 81 452 LDA VARNAM
9615: 30 0A 453 BMI INTAI
9616: A5 82 454 LDA VARNAM+1
9617: 30 0C 455 BMI TYPERR
9618: A9 01 456 LDA #1
9619: 8D FF 95 457 STA TYPFLG
9620: 85 458 RTS
9621: 459 * INTAI LDA #2
9622: A9 02 460 STA TYPFLG
9623: 8D FF 95 461

```

```

95CB: 60 RTS
95CC: AD 54 C0 PAGE1 LDA PAGE1
95CD: 4C C9 DE JMP SNERR
95CE: * Wertzuweisung der BASIC-Variablen mit Farbcode
95CF: * im A-Register
95D0: 48 470 GIVBACK PHA ;Farbwert retten
95D1: AD FF 95 471 LDA TYPFLG
95D2: C9 01 472 CMP #1
95D3: F0 0B 473 BEQ REAL1 ;Real-Var.
95D4: 68 474 PLA
95D5: A0 01 475 LDA #1
95D6: 91 CE 476 STA (POINTER),Y ;High-Byte
95D7: A9 00 477 LDA #0
95D8: 88 478 DEY
95D9: 91 CE 479 STA (POINTER),Y ;Low-Byte
95DA: 60 480 RTS
95DB: 68 481 PLA
95DC: 483 REAL1 TAY
95DD: A8 482 TAY
95DE: 20 01 E3 483 JSR SNGFLT ;YREG -> Float -> FAC
95DF: 00 00 484 LDA #0
95E0: B9 9D 00 485 LDA FAC,Y ;FAC -> Var.
95E1: 91 CE 486 STA (POINTER),Y
95E2: C8 487 INY
95E3: C0 05 488 CPY #5
95E4: D0 F6 489 BNE REAL2
95E5: A5 9E 490 LDA FAC+1
95E6: A0 01 491 LDA #1
95E7: 29 7F 492 LDY #1
95E8: 91 CE 493 AND #%01111111 ;SIGN auf +
95E9: 91 CE 494 STA (POINTER),Y
95EA: 60 495 RTS
95EB: 00 496 * TYPFLG HEX 00
95EC: 00 497
95ED: 00 498

```

742 Bytes

### DOUBLE.LORES.DEMO

```

10 PRINT CHR$(21): PRINT CHR$(4)"BRUN DOUBLE.LORES"
15 REM DOUBLE.LORES.DEMO zeichnet eine Kugel
20 TEXT : GOSUB 85
25 XE = 79:YE = 47:A = XE / 2:B = YE / 2:F1 = 7:F2 = 23
30 FOR C = 1 TO 15: COLOR=C:R = C + C
35 FOR X = A - R TO A + R: IF X < 0 OR X > YE GOTO 60
40 H = B + SQR (H - (X - A) * (X - A)): IF H < = 0 GOTO 60
45 Y1 = B + SQR (H) - F1:Y2 = B - SQR (H) + F1
50 IF Y1 > 0 AND Y1 < = YE THEN & PLOT X,Y1
55 IF Y2 > 0 AND Y2 < = YE THEN & PLOT X,Y2
60 NEXT X
65 NEXT C
70 GET X$
75 & TEXT : HOME : END
80 REM Umschaltung auf Full-Screen
85 & GR : POKE 49234,0
86 POKE 49236,0: CALL 63538
87 POKE 49237,0: CALL 63538
90 RETURN

```

# BUCH-SHOP

## Betriebssystem CP/M

Vom Monitorprogramm zum Mehrbenutzersystem.  
Von Jürgen Plate.  
1984, 351 Seiten, 30 Abb.,  
3 Tab., geb., DM 56,-



Das Buch beschreibt ausführlich die Kommandos, ihre genaue Syntax und die einzelnen Teilprogramme von CP/M wie BIOS (systemspezifischer Teil), ED (Editor), ASM (Assembler, inklusive einer Beschreibung des 8080-Befehlssatzes), SYSGEN und STAT. Der Beschreibung von CP/M ist das Listing eines komfortablen Monitorprogramms für Z-80-Computer vorangestellt, das eine elementare Programmierung auf Maschinenebene erlaubt, solange man CP/M noch nicht geladen hat. Das kann z. B. zur Fehlersuche sehr nützlich sein. Am Schluß des Buches findet sich auch eine Kurzbeschreibung der Multitasking-/Multiuser-Betriebssysteme.

## Das Buch zum Apple II

von Erich Esders  
1985, 210 S., 119 Abb., geb.,  
DM 54,-



Wenn hier vom Apple II gesprochen wird, so gilt das auch für den IIplus, den IIeuroplus und die IIe-Versionen sowie für den ganzen „Apple-Nachbau“. Das Buch ist ein Wegweiser durch diesen Rechner, um mit ihm schneller und effektiver zu arbeiten. Es geht hier weniger um das elementare Programmieren des

Rechners, sondern um Assemblerprogramme, die extensiv Monitor-ROM-Subroutinen benutzen. Diese hat der Autor nach Sachgebieten geordnet, z. B. Mathematik, Graphik, String-Bearbeitung + Disassembler-Listings und diese wiederum mit Erklärungen und Applikationen komplettiert. Eine ausreichende Dokumentation ist dabei immer gewährleistet. Sie geht schrittweise vor, von der Aufgabenstellung über die Programmentwicklung bis zum lauffähigen Maschinenprogramm. Die angebotenen Beispiele sind ausbaufähig und lassen der eigenen Kreativität reichlichen Spielraum. Viele neuartige Tips und Tricks wird auch der beschlagene Apple-Benutzer begrüßen.

Aus dem Inhalt:  
Der Mikroprozessor des APPLE II. Der APPLE II und seine Speicheraufteilung. APPELSOFT und seine Arbeitsspeicher-Bereiche. Der MICROSOFT-Basic-Interpreter: Die Zeichen-Lese-Routine. Interpretierer und Lokalisierer. Handler-Routinen. BASIC/Maschinensprache-Interfaces. DISAS-Generator. Unterprogramme im APPELSOFT-Basic-Interpreter: Softschalter und -Flags. Ausdrucks-Interpreter. Low-Resolution-Graphik. Fehler-Behandlung. Applikationen: Arithmetik-Demonstration „FP-CALC“. Hex-Dumps der Applikationen. BASIC-Monitor BASMON/D: Vorstellung der neuen Kommandos. Das Programm „BASMON/D“. Implementierung und Laufbeispiele. BASIC-Interpreter-Vergleich APPLE II – Commodore 64: Arithmetik-Demonstration „FP-CALC/64“. Listen: Die Token des APPELSOFT-Basic.

## Apple II ROM Listing

von Matthias Buck  
1984, 116 S., Kart., DM 59,-



Das deutsche Apple-II-ROM-Listing ist da! Einleitung zum prinzipiellen Ablauf des Applesoftinterpreters:

- Aufbau und Verarbeitung

der/des Programmtextes – Variablen-tabelle – String-space – Fließkommaformate – Basicstacks (GDSUB, FOR-NEXT, ...)

- Beschreibung der wichtigsten Unterprogramme, z. B. Variablensuche, Garbage collection, Ausdrucksauswertung, CHRGET, ...
  - Vollständig disassemblierte und sehr ausführlich deutsch kommentierte Auflistung des Applesoft-BASIC-Interpreters
  - Übersichtliche Auflistung aller vom Interpreter benutzten RAM-Zellen mit allen Verwendungszwecken
  - Über 150 ausführlich dokumentierte Unterprogramme:
    - Funktion
    - Ein/Ausgabeparameter
- Auch für Apple-IIe und c und Kompatible!

## Apple II Pascal

Eine praktische Anleitung  
von Arthur Luehrmann und  
Herbert Peckham  
1982, 544 S., kart., DM 59,-



Dieses Buch ist unentbehrlich für alle, die die Programmiersprache PASCAL lernen wollen und Zugang zu einem Apple Computer haben. Sie benötigen keinerlei Vorkenntnisse, sondern lernen an Hand von Beispielen und Übungen, wie man selbst PASCAL-Programme entwickelt und sie austestet und werden allmählich von Kapitel zu Kapitel vertrauter im Umgang mit dem Apple Computer.

## Start mit Apple-Logo für II, IIe und IIc

Das kleine Logo-Einmaleins: Grafik • Text • Musik  
Von D. Senftleben  
1985, 222 Seiten, DM 35,-

Viele Mikrocomputer-Hersteller bieten für ihre Geräte neben BASIC und anderen Programmiersprachen zunehmend auch Logo an. Durch ihre Benutzerfreundlichkeit hat diese Sprache bereits viele Freunde im Ausbildungs- und Freizeitbereich gefunden. Dabei ist Logo eine mächtige Sprache, die auch dem anspruchsvollen Anwender kaum Wünsche offenläßt. Mittels Schildkrötengrafik wird das kleine Logo-Einmaleins in 12 Lektionen entwickelt. Große Bildschirmfotos begleiten den Leser durch die Lektionen. Das Buch verlangt aktive Mitarbeit. Es hat seinen Platz neben dem Computer und gibt Hilfen und Anregungen für eigenes Forschen. Dank des bausteinerorientierten Konzepts kann jeder seine eigenen Teilbausteine erzeugen und sie zu neuen Blöcken zusammenfügen. Neben dem Einmaleins werden neue Einsatzbereiche für den Einsteiger erschlossen. Musik und Sound fehlen nicht.



In diesem Buch werden die beiden offiziellen Logo-Produkte der Firma Apple für die Rechnerfamilie Apple II, IIe und IIc behandelt und deren Unterschiede verdeutlicht. Weiterhin sind sämtliche Apple-Logo-Vokabeln übersichtlich zusammengestellt. Dieses Buch ist ideal zum problemlosen und vergnüglichen Start in die Apple-Logo-Welt.

## Apple II Anwenderhandbuch

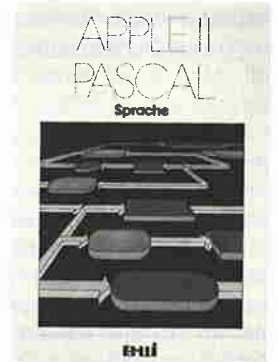
von Lon Poole  
1982, 450 S., zahlr. Abb.,  
kart., DM 56,-



Auch für diesen Computer haben wir den richtigen Leitfaden. Er erspart Ihnen zeitraubendes und nutzloses Suchen nach der wirklich verwendbaren Dokumentation für Ihren Computer. Das Anwenderhandbuch beschreibt zum einen den beliebigen Apple II-Computer als solchen und gibt zum anderen ausführlich Auskunft über die normalen Peripheriebausteine und Zubehör einschließlich Disk-Laufwerken und Drucker. Mit Hilfe dieses Buches werden Sie Ihren Apple II erfolgreich einsetzen, denn der Informationsgehalt geht weit über das hinaus, was herstellerseitig an Literatur angeboten wird. Sie lernen BASIC auf zwei verschiedene Arten zu verwenden. Wie man den Gebrauch von Klang, Farbe und Grafik zum Optimum führt. Sie erhalten Tips für fortgeschrittene Programmierung. Sie erfahren die Verwendung des Maschinensprachen-Monitors u. v. m. Mit dem Apple II-Anwenderhandbuch werden Ihnen alle Möglichkeiten eröffnet, die in diesem Computer stecken.

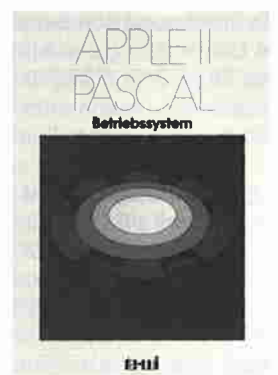
## Apple II Pascal

Sprache  
1985, 197 S., DM 39,-



## Apple II Pascal

Betriebssystem  
1985, 256 S., DM 49,-



# Adreßverwaltungsprogramm mit dBASE II

von Ing. (grad.) Ernst Fischer

Ein gutes Adreßverwaltungsprogramm (künftig Adv genannt) sollte mindestens folgende Anforderungen erfüllen:

Das Programm soll anwenderfreundlich sein; es kommt also nur eine **Menütechnik** in Frage. Personaldaten müssen leicht veränderbar sein; es sollte eine **Suchfunktion** nach Vorname, Familienname und nach Bemerkungen existieren. Vorteil: Man findet auch Personen wieder, deren Namen sich durch Heirat verändert haben. Die Suchfunktion nach Bemerkungen sollte es ermöglichen, auch Teilstrings innerhalb der Bemerkung zu finden. Von einem Adv-Programm erwartet man nicht nur, daß es für die gespeicherten Adressen **Adreßaufkleber** und Absender druckt, sondern auch ein **Adreßverzeichnis** im Taschenkalenderformat ausgibt. Schließlich müssen Datensätze leicht, aber nicht versehentlich gelöscht werden können. Ein gutes Adv sollte nur über ein **Passwort** benutzbar sein, um es vor nichtautorisierten Personen zu schützen. Es interessiert den Benutzer auch, wieviele Datensätze gespeichert sind und wann der letzte Zugriff war. Diese Informationen sollten bei Beginn vom Adv ausgegeben werden. All diese Forderungen werden von diesem Programm erfüllt.

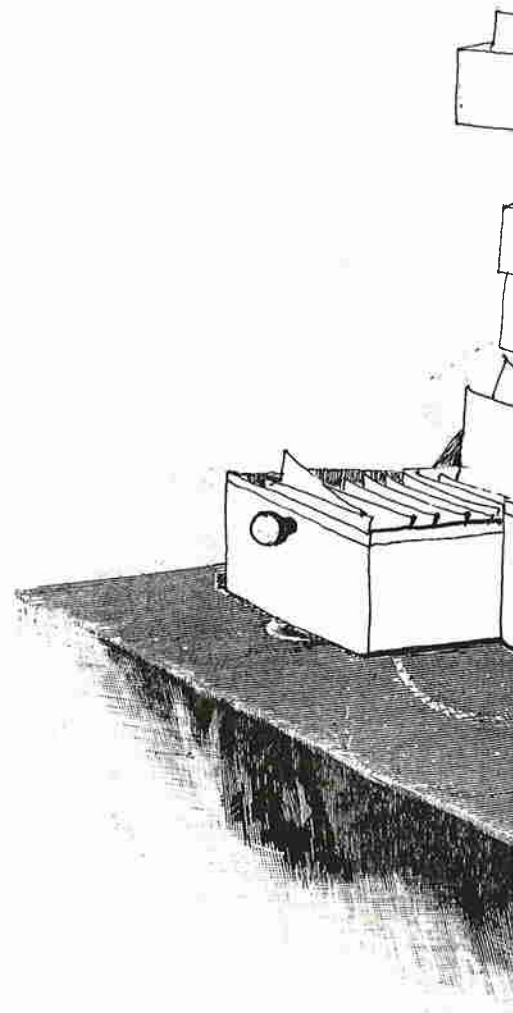
**Als Grundlage dient die relationale Datenbank dBASE.** Einer der wesentlichsten Vorteile von dBASE ist die Flexibilität, bedingt durch eine eigene Programmiersprache. Mit ihr läßt sich praktisch jedes Problem lösen, wie im folgenden noch gezeigt wird. Für eine sinnvolle Anwendung

ist folgende Gerätekonfiguration erforderlich: Ein Apple II/II+/IIe oder kompatible Nachbauten, zwei Diskettenlaufwerke, ein Drucker und eine Z80-Karte (läuft unter CP/M). Natürlich muß man im Besitz der relationalen Datenbank dBASE sein. Leider können Besitzer eines Apple IIc die Vorteile dieses Programms wegen der fehlenden Slots nur mit einer teuren Zusatzkarte nutzen. Die Druckersteuerzeichen in den Unterprogrammen beziehen sich auf **Epson**-Drucker.

Wichtig bei der Anwendung: Zu beachten ist bei der Benutzung des Adv lediglich, daß man das Programm nur über die Option 9 des Hauptmenüs verläßt. Nur so ist gewährleistet, daß die benutzten Dateien wieder geschlossen werden. Hält man sich nicht an diese Selbstverständlichkeit, kann dies zu Datenverlusten führen.

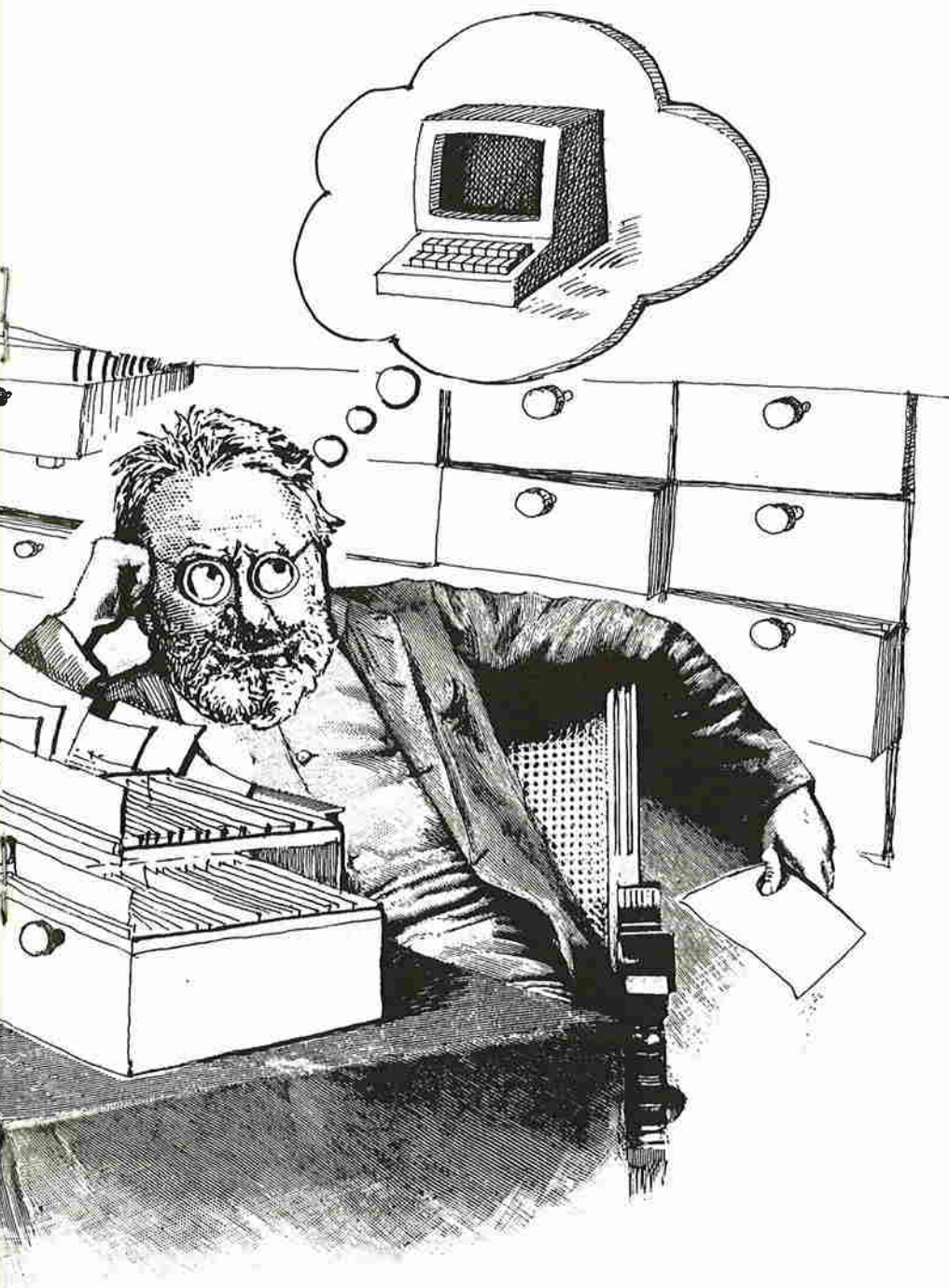
## Aufbau des Adreßverwaltungsprogramms

Wie **Bild 1** zeigt, wird der Benutzer über den aktuellen Zustand des Adv informiert. Es zeigt an, wieviele Adressen gespeichert sind und an welchem Datum das Adv zum letzten Mal benutzt wurde. Schließlich wird der Benutzer aufgefordert, das aktuelle Tagesdatum einzugeben. Bei einem Return wird das alte Datum übernommen. Nachdem diese Hürde genommen ist, wird der Benutzer aufgefordert, seine Zugriffsberechtigung durch ein Passwort nachzuweisen. Das Passwort wird dabei aus Sicherheitsgründen nicht am Bild-





# ngs-



schirm angezeigt. Bei richtiger Eingabe erschließt sich nun erst das Hauptmenü; man kann über die gespeicherten Daten verfügen.

### Auflistung und Funktion der einzelnen Programme (vgl. Bild 2)

Die Programme in Laufwerk A: haben die Aufgabe, menügesteuert auf die Datenbank zugreifen zu können. Außerdem befinden sich auf A: die Ein- und Ausgabemaske, welche für die Dateneingabe, Datenausgabe und Datenveränderung genutzt werden. Die drei Indexdateien gestatten einen sehr schnellen Zugriff auf die Daten (maximal 2 Sekunden unabhängig von der Datengröße). Mit ihnen kann man das indizierte Feld alphabetisch ausgeben. Indizierung hat gegenüber Sortierung den Vorteil, daß die Daten physisch nicht verändert werden und auch weniger Zeit und Speicherplatz beansprucht wird. Das Indexregister wird auf Laufwerk A: angelegt, weil hier noch Speicherplatz auf der Diskette zur Verfügung steht, der bei großen Datenmengen bei Laufwerk B: fehlen würde. In der Memorydatei werden schließlich die aktuellen Daten des Adv wie die Anzahl der gespeicherten Records und das letzte Zugriffsdatum abgelegt. In Laufwerk B: befindet sich ausschließlich die Adv Datenbank, deren Struktur später erklärt wird.

### Aufbau der Datenbank

Das Adv sollte so universell wie möglich sein und dabei alle Informationen liefern, die beruflich und auch privat interessant sind. Sollten für das Adv bestimmte Daten, wie Geburtsdatum oder Vorname der Ehefrau zu intim sein, weil man die Person nicht so genau kennt, so läßt man einfach dieses Feld frei. Die Datenspeicherung erfolgt ausschließlich auf dem Laufwerk B:.. Somit ist genügend Platz auf der Diskette. Alle Kommandodateien, Indexregister und der Memoryspeicher sind auf Laufwerk A:.. Für die Felder FAM, VOR und VORE ist jeweils ein Indexregister angelegt; das spart Zeit beim Suchen und gestattet auf einfache Weise das alphabetische Ordnen.

Die Datenbank wird folgendermaßen angelegt: Man startet dBASE im Laufwerk A: mit der Eingabe von „dbase“ <RETURN> (das CP/M Betriebssystem muß natürlich auf der dBASE-Diskette sein). Nach der Eingabe des Tagesdatums oder einem Return meldet sich dBASE mit einem Punkt

als Promptzeichen. Für die Generierung des Adv geben Sie den dBASE-Befehl CREATE B:ADRESSEN ein. Daraufhin erfragt sich dBASE die Struktur der Datensätze. Nachdem Sie das 17. Feld eingegeben haben, schließen Sie mit RETURN ab. Auf die Frage, ob Sie die Daten jetzt sofort eingeben wollen (INPUT DATA NOW?) antworten Sie mit „N“ (No). Damit liegt die Struktur des Adv in Laufwerk B: fest. Vergleiche hierzu **Bild 3**.

### Eingabe der Kommandodateien

Die Eingabe sollte nicht mit dem dBASE-Editor mit Hilfe des MODIFY-COMMAND-Befehls eingegeben werden. Es hat sich gezeigt, daß ab einer bestimmten Länge der Programme der Editor verrückt spielt. Statt dessen sollte die Programmeingabe mit WordStar erfolgen. Außerdem hat man dabei mehr Komfort.

(Anm. der Red.: Wegen des großen Umfangs der Kommandodateien sind alle erforderlichen CMD-Files mit Ausnahme eines Beispiels {**Bild 5**} nur auf der Peeker-Sammeldiskette enthalten. Da diese Diskette im DOS-Format beschrieben ist, transferieren Sie die Files mit dem Programm **APDOS** der CP/M-Systemdiskette von DOS nach CP/M. Danach liest dBASE diese Files einwandfrei.)

### Beschreibung der Programmteile

#### START.CMD (Programmbeschreibung)

Das Programm gibt Informationen über die Datenbank (letztes Zugriffsdatum, Anzahl der Records usw.). Diese bezieht es zum Teil aus dem Memoryspeicher von dBASE. Die Datumseingabe wird auf gültige Werte überprüft; erst dann kann im Programm fortgefahren werden. Es erfolgt eine Überprüfung des eingegebenen Passwortes. Wünschen Sie ein anderes Passwort, so ersetzen Sie „PEEKER“ durch das Wort Ihrer Wahl. Wird später beim Gebrauch ein falsches Passwort eingegeben, so löscht dBASE alle bis dahin gemachten Eingaben vom nichtautorisierten Benutzer und kehrt zum CP/M-Betriebssystem zurück. Stimmt das eingegebene Passwort mit dem gespeicherten überein, wird das Hauptmenü aufgerufen.

#### HMENUE.CMD (Programmbeschreibung)

Das Hauptmenü gestattet den Zugriff auf die einzelnen Unterprogramme. Wird Option 9 (Ende) gewählt, so werden zuerst alle Speichervariablen mit „RELEASE ALL“ gelöscht. Dann wird die Datenbank mit „USE B:ADRESSEN“ eröffnet und der

Zeiger auf den letzten Record gesetzt. Die Nummer dieses Records wird im Variablenpeicher „R:NR“ in den Memoryspeicher geladen. Danach wird mit „CLEAR“ die Datenbank geschlossen. „CANCEL“ verläßt das Hauptmenü und das dBASE-Promptzeichen erscheint. Ersetzt man „CANCEL“ mit „QUIT“, so kehrt man zum CP/M-Betriebssystem zurück.

#### AUFNAHME.CMD (Programmbeschreibung)

Zuerst wird die Adv-Datenbank mit „USE B:ADRESSEN“ eröffnet und der Zeiger an das Datenende gesetzt. Mit „APPEND BLANK“ wird ein leerer Datensatz angehängt, der dann durch den Aufruf der Aufnahmemaske gefüllt wird. Die WHILE-Schleife wird solange durchlaufen, bis man die Frage „Weitere Eintragungen?“ mit „N“ beantwortet. Hat man die Eingabe durch „N“ abgebrochen, so werden drei Indexdateien angelegt; eine für den Familiennamen, eine für den Vornamen der Eingabeperson und eine für den Vornamen von Ehefrau/Ehemann. Mit „CLEAR“ wird die Datenbank geschlossen.

#### AUFMASKE.CMD (Programmbeschreibung)

Die Aufnahmemaske dient zur einfachen und übersichtlichen Eingabe der Personaldaten. Dieser Programmbaustein wird auch bei Veränderung der Personaldaten genutzt. Das Programm erzeugt den Bildschirm Ausdruck von **Bild 4**.

#### AUSGABE.CMD (Programmbeschreibung)

Die Ausgabemaske (AM) ist im Prinzip genauso wie die Eingabemaske (EM) aufgebaut. Der Unterschied ist folgender: Bei der EM mußten ja die Daten in die Speichervariablen eingelesen werden; deshalb die „GET“-Befehle. Bei der AM werden die Speichervariablen aber auf den Bildschirm gegeben. Hier also keine „GET“-Anweisung, sondern „SAY“-Anweisungen. Der Cursor bleibt sofort auf der letzten Zeile (hinter J/N) stehen; man braucht sich nicht mit Return bis zur letzten Zeile vorarbeiten.

#### SUCH.CMD (Suchprogramm)

Das Suchprogramm sucht, wie sein Name es vermuten läßt, nach einem bestimmten Datensatz, wenn der Familienname oder ein Teil davon bekannt ist. Die WHILE-Schleife wird solange durchlaufen, wie der vorhandene Datensatz 0 und somit noch nicht gefunden ist. Ist der Zeiger am Ende der Datei, erscheint die Meldung, daß es diese Person nicht gibt. Man hat die Mög-

lichkeit, mit Return die Suche abzubrechen oder, nachdem man den Namen der gesuchten Person neu eingegeben hat, einen neuen Suchvorgang zu starten. SUCH.CMD wird in folgenden Kommandodateien verwendet:

- EDITFNAM.CMD
- SUCHFNAM.CMD
- SCHREIBA.CMD
- LOESCH.CMD

#### EDITFNAM.CMD (Programmbeschreibung)

Dieses Programm erlaubt Änderungen von Datensätzen. Nach der Eingabe des Familiennamens (optional auch zusätzlich des Vornamens) sucht dBASE in der Indexdatei den gewünschten Namen. Verläuft die Suche negativ, erscheint eine entsprechende Meldung. Der Benutzer hat die Möglichkeit, beliebig oft die Namens-eingabe zu wiederholen oder durch ein Return zum Hauptmenü zurückzukehren. Wurde auch der Vorname eingegeben, so erfolgt nun mit „LOCATE“ die Suche nach dem Datensatz, der beide Kriterien erfüllt (Vor- und Familienname). Die „FIND“-Anweisung dient dabei zum schnellen Finden des Familiennamens. Da davon auszugehen ist, daß verschiedene Personen den gleichen Familiennamen haben, ist es zweckmäßig, auch den Vornamen einzugeben. Mit dem LOCATE-Befehl wird dann die Person sicher gefunden. Die Aufnahmemaske erlaubt dabei eine bequeme Veränderung der Daten.

#### SUCHFNAM.CMD (Programmbeschreibung)

Will man sich die Daten einer Person komplett am Bildschirm anschauen, so ruft man dieses Kommandoprogramm auf. Nachdem Sie den Familiennamen der gewünschten Person eingegeben haben, sucht das Unterprogramm SUCH.CMD den Datensatz. Wird er nicht gefunden, können sie mit Return abbrechen oder einen erneuten Suchvorgang einleiten. Die Daten werden mit der Ausgabemaske am Bildschirm angezeigt. Beantworten Sie die Frage „Ist dies die gewünschte Person (J/N)“ mit N(ein), so wird mit LOCATE die nächste Person gesucht, auf welche das Suchkriterium zutrifft. Dieses Spiel geht so lange, bis die Person gefunden ist oder der Datenzeiger am Dateiende angelangt ist.

#### SUCHVNAME.CMD (Programmbeschreibung)

Manchmal weiß man von einer Person nur den Vornamen, weil der Familienname

sich durch eine Heirat geändert hat. Das Programm ist ähnlich aufgebaut wie SUCHFNAME.COM. Wird der Vorname nicht in der Indexdatei Beta gefunden, so wird in der Indexdatei Gamma weitergesucht. Hier sind die Vornamen der Ehepartner indiziert. Verläuft die Suche auch hier ergebnislos, so erfolgt die Meldung „Den Vornamen XXX gibt es nicht!“. Mit Return kann man die Suche abbrechen oder durch eine erneute Eingabe den Suchvorgang wiederholen. Ist die angezeigte Person nicht die richtige, so werden mit LOCATE nacheinander alle Personen mit dem entsprechenden Vornamen ausgegeben, bis der EOF-Marker erreicht ist.

**SUCHBEME.COM** (Programmbeschreibung)

Oft kommt es vor, daß man von einer Person sowohl Familien- als auch Vornamen vergessen hat. Dann helfen bezüglich der Person gemachte Bemerkungen weiter. Das Programm findet auch Teilstrings aus der Bemerkung. Ein Beispiel: Man hat als Bemerkung „Mitglied beim Computerarbeitskreis“ eingegeben. Als Teilstring genügt „arbeitskreis“. Es werden alle Personen nacheinander angezeigt, die bei Bemerkungen den Teilstring „arbeitskreis“ haben. Wird der Teilstring nicht gefunden, so kann man wieder mit Return die Suche abbrechen oder ein neues Suchkriterium eingeben.

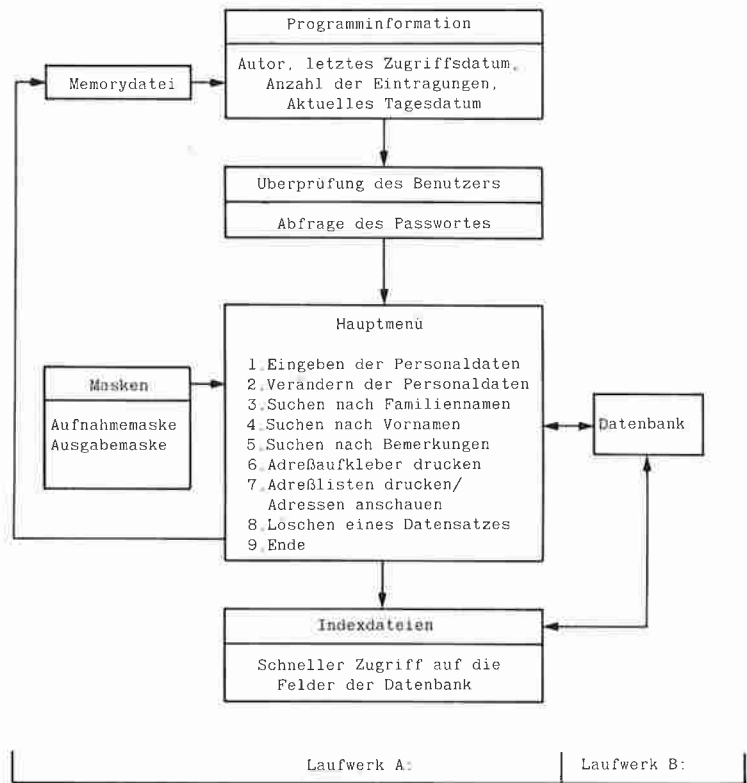
**SCHREIBA.COM** (Programmbeschreibung)

Zu einer Adreßverwaltung gehört auch ein Programm, das Adressen- und Absenderaufkleber drucken kann. Die gewünschte Person wird mit dem schon bekannten Suchprogramm SUCH.COM gefunden. Wie man aus dem Anredenmenü sieht, hat man alle erdenklichen Optionen. Wählt man Option-Nr.1, so kann man für sich Absenderaufkleber drucken lassen. Option-Nr.7 gestattet individuelle Anreden wie Firma o.ä. Hat man sich für eine der sieben Möglichkeiten entschieden, so erscheint die Adresse richtig formatiert auf dem Bildschirm. Hat man sich zum Ausdruck entschieden, wird man nach der Anzahl der zu druckenden Exemplare gefragt und gebeten, den Drucker einzuschalten. Nach dem Justieren der Aufkleber kann nach einem Return der Druck beginnen. Die Druckersteuerzeichen sind für den **Epson FX-80** gedacht.

**SCHREIBL.COM** (Programmbeschreibung)

Eine Adreßverwaltung taugt nur etwas, wenn man die gespeicherten Adressen

**Bild 1:** Aufbau des Adreßverwaltungsprogramms



**Bild 2:** Auflistung der Unterprogramme des Adv in Laufwerk A:

Programmname	Funktion des Programmes
AD-START.COM	--> Startprogramm, informiert über das Programm, Recordanzahl, letztes Zugriffsdatum usw.
HMENU.COM	--> Erlaubt den Zugriff auf die einzelnen Unterprogramme.
AUFNAHME.COM	--> Dient zur Dateneingabe; es wird eine Eingabemaske benutzt.
AUFMASKE.COM	--> Maske zur Dateneingabe und Datenveränderung.
AUSMASKE.COM	--> Maske zur Datenausgabe.
EDITFNAME.COM	--> Erlaubt die Veränderung der Personaldaten; es wird die Maske zur Dateneingabe benutzt.
SUCHFNAME.COM	--> Erlaubt die Suche nach einer bestimmten Person; Als Suchkriterium dient der Familienname.
SUCHVNAME.COM	--> Erlaubt die Suche nach einer bestimmten Person; Als Suchkriterium dient der Vorname.
SUCHBEME.COM	--> Erlaubt die Suche nach einer Bemerkung oder einem Teil in der Bemerkung.
SCHREIBA.COM	--> Schreibt Adressenaufkleber.
SCHREIBL.COM	--> Schreibt Adressenlisten im Taschenkalenderformat und listet auf dem Bildschirm die Records auf.
LOESCH.COM	--> Löscht die Personaldaten einer Person.
SUCH.COM	--> Sucht nach gewünschten Datensätzen.

**Bild 3:** Struktur der Datenbank

Feld	Name	Bezeichnung	Type	Feldbreite
1	FAM	Familienname	C	15
2	VOR	Vorname	C	14
3	PLZ	Postleitzahl	C	4
4	ORT	Wohnort	C	15
5	STR	Straße	C	25
6	GEBD	Geburtsdatum	C	8
7	BEM	Bemerkung	C	50
8	TNRP	Privattelefonnr.	C	6
9	VWP	Vorwahl Privat	C	6
10	VWAL	Auslandsvorwahl	C	6



immer verfügbar hat in Form einer Hartkopie. Verfügbar heißt, immer und überall greifbar. Dieser Forderung kommt das Programm in besonderer Weise nach. Der Ausdruck erfolgt nämlich im Taschenkalenderformat; auf einer Größe von 8 cm x 14,5 cm bekommt man einen Ausdruck, den man in gebundener Form in jeden handelsüblichen Taschenkalender einschieben kann. Zu diesem Zweck schneidet man sich einzelne Blätter von der Größe 16 cm x 14,5 cm, die man in der Mitte faltet und beidseitig bedrucken läßt. Die Papierende-Erkennung des Druckers wird automatisch abgeschaltet. Vor dem Blattende wird man aufgefordert, ein neues Blatt einzuspannen. Es erfolgt dabei eine automatische Zeilenberechnung, so daß eine Adresse immer auf einem Blatt endet und nicht ein Teil auf die nächste Seite übernommen werden muß. Die Datenausgabe wird so formatiert, daß nicht vorhandene Personalien auch nicht ausgegeben werden. Beispiel: Es ist keine Bemerkung abgespeichert; also wird diese Zeile aufgerückt und die Abkürzung „Bemerk.“ erscheint nicht. Oder: Der Vorname ist nicht bekannt; der Familienname wird linksbündig geschrieben.

Neben einer Hartkopie gibt dieses Programm auch eine geordnete Übersicht der gespeicherten Daten am Bildschirm aus. Die Formatierung erfolgt in der gleichen Weise wie beim Drucken.

**LOESCH.CMD** (Programmbeschreibung)  
 Ab und zu muß man seine Adreßdatei bereinigen. Man möchte bestimmte Personen aus der Datei löschen. Das Programm fragt nach dem Familiennamen der zu löschenden Person. Das Suchprogramm SUCH.CMD verfährt nach der uns schon bekannten Weise und zeigt uns, anhand der Ausgabemaske, die erste Person mit diesem Namen. Ist dies nicht die gewünschte Person, so wird mit LOCATE die nächste Person präsentiert, auf die das Suchkriterium zutrifft. Das Spiel geht solange, bis entweder die gesuchte Person gefunden ist, oder der Zeiger an der EOF-Marke angekommen ist. Ist die Suche positiv verlaufen, erscheint eine deutliche Warnung, daß die Person mit den folgenden Daten gelöscht wird. Man hat jetzt die letzte Möglichkeit, den Löschvorgang abzubrechen. In diesem Fall antwortet man auf die Frage „Ist dies die gewünschte Person zum Löschen (J/N)“ mit „N“. Antwortet man mit „J“, so wird einem noch einmal der Name der Person gezeigt, von der soeben alle Daten gelöscht wurden.

11	VWD	Vorwahl Dienst	C	6
12	TNRD	Diensttelefonnr.	C	8
13	NAT	Nationalität (KFZ)	C	2
14	VORE	Vorname der Ehefrau	C	14
15	GEBDE	Geburtsdatum Ehefrau	C	8
16	VWDE	Vorwahl Dienst Ehefrau	C	6
17	TNRDE	Diensttelefonnr. der Ehefrau	C	8

Bild 4: Maske zur Aufnahme der Adreßdaten

```

+-----+
|
|  Bitte geben Sie die Daten für das Adreßregister ein...
|
+-----+
| Fam_Name:                Vorname:                Geb.Datum:
|
| Ehefrau..... Vorname:                Geb.Datum:
|
| PLZ:    Wohnort:                Straße u.Nr.:
|
+-----+
| Land (Intern.KFZ-Zeichen als Abkürz.):
|
+-----+
| Telefon Priv. Vorwahl:    Rufnr.:    Auslandsvorwahl:
|
| Telefon Die. Vorwahl:    Rufnr.:
|
| Ehefrau... Telefon Dienstlich Vorwahl:    Rufnr.:
+-----+
| Bemerkung:
+-----+
| Weitere Eintragungen ? (J/N):
+-----+

```

Bild 5: Beispiel für die erste Command-Datei (AD-START.CMD)  
 (Die übrigen CMD-Files befinden sich auf der Peeker-Sammeldisk)

```

* DO START.CMD (Startet das Adv)
ERASE
SET TALK OFF
RESTORE FROM MEMORY
STORE CHR (15) TO INVERSE
STORE " " TO ZGBER
STORE F TO OK
$ 23,0 SAY INVERSE + "ADRESSEN" S T A R T " +
" "
$ 6,21 SAY " A D R E S S E N V E R W A L T U N G"
$ 8,21 SAY "APPLE //e Ver. 1.0 8/84"
$ 10,21 SAY "AUTOR : ERNST FISCHER"
$ 15,9 SAY "Letzter Zugriff : " + DATE +
" Anzahl Records : "
$ 15,68 SAY STR(R:NR,3)
DO WHILE .NOT. OK
$ 17,9 SAY "Bitte geben Sie das Datum ein <TT/MM/JJ> "
GET DATE PICTURE '99/99/99'
READ
IF VAL$(DATE,1,2) > 31;
.OR. VAL$(DATE,1,2) < 1;
.OR. VAL$(DATE,4,2) > 12;
.OR. VAL$(DATE,4,2) < 1;
.OR. VAL$(DATE,7,2) < 83
? CHR (7)
ELSE
STORE T TO OK
ENDIF
ENDDO
ERASE
STORE $(DATE,1,2) + " " + $(DATE,4,2) + " " + $(DATE,7,2) TO DATUM
SET DATE TO &DATUM
$ 23,0 SAY INVERSE + "ADRESSEN" S T A R T " +
" DATUM : " + DATE()
$ 17,5 SAY "Bitte Passwort eingeben"
SET CONSOLE OFF
ACCEPT ' ' TO ZGBER
SET CONSOLE ON
ERASE
RELEASE OK, DATUM, DATE
IF ZGBER < > "PEEKER"
RELEASE ALL
ERASE
QUIT
ENDIF
DO HMENU
*Ende von AD-START

```



# Wordstar druckt internationale Zeichensätze

Frei definierte Sonderzeichen auf dem FX-80 nutzen

von Dipl.-Ing. H. A. Rohrbacher

Normalerweise drucken die Epson-Matrixdrucker der MX- oder FX-Serien den mit ihren DIP-Schaltern eingestellten Vorzugszeichensatz aus. In aller Regel ist dies der deutsche Satz mit den Umlauten, dem ß und dem Paragraphzeichen. Dies geschieht völlig unabhängig von der Darstellung der Sonderzeichen auf dem Bildschirm.

Nun kommt es häufig vor, daß man entweder in einer fremden Sprache korrespondieren oder in deutschen Texten die eckigen oder geschweiften Klammern verwenden möchte, die aber beim deutsch eingestellten Drucker nicht ohne weiteres erreichbar sind. Hier bietet der Drucker die Möglichkeit eines Software-Schalters, der die Umschaltung auf einen zweiten Sprachensatz veranlaßt. Beim Epson-Drucker erfolgt die Auswahl des Zeichensatzes über ESC R (n), also in Applesoft durch PRINT CHR\$(27) "R"; CHR\$(n), wobei „n“ die Zeichensatz-Nummer ist (ASCII: n = 0, Französisch: n = 1, Deutsch: n = 2 usw.).

^A = ASCII EIN		^N = DEUTSCH EIN	
{ = ä	= ö	} = ü	
[ = x	\ = ä	] = ü	
~ = ß	@ = s	' = '	
^E -> ^R	Engschrift ETR/AUS äöüß&os = ({}~[])		
^W -> ^Q	weit AU=[ ]		
^W^E -> ^R^Q	Sperrschrift		
^D -> ^D	doppelt gedruckt		
^B -> ^B	fett gedruckt öö=!!!		
^W^D -> ^Q^D	weit t+doppelt		
^T -> ^T	Tiefstellen äöü=()		
^T^E -> ^T^R	Tief+eng äöü=()&os		
^Y -> ^Y	hochstellen äöü=()		
^Y^W^E -> ^Y^Q^R	Sperr+hochstellen		
^Y^W -> ^Y^Q	weit t+hoch		

Tabelle 2

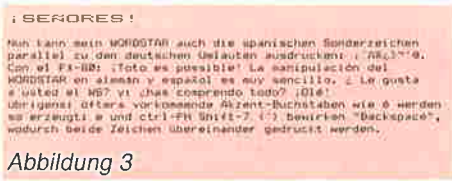
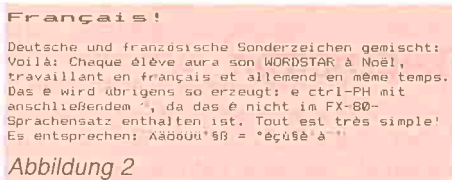
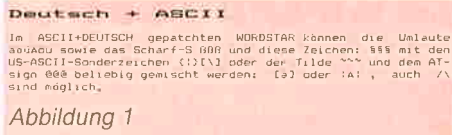
Adresse:	Adr.†	Ctrl-P	Inhalte	Bedeutung
TRMINI:	0292		04 14 48 1A 33	CTRL-Z3 -> deutsche Zeichen
DEL3:	02D1		01 (ex: 19)	mittellange Warteschleife AUS
DEL4:	02D2		01 (ex: 40)	lange Warteschleife AUS
DEL5:	02D3		00 (ex: 09)	Ctrl-delay AUS
PALT:	06B5:	↑A	03 1B 52 00	ESC-Sequ.-> ASCII-Zeichen EIN
PSTD:	06BA:	↑N	03 1B 52 02	ESC-Sequ.-> deutsche Zch. EIN
USR1:	06C9:	↑Q	03 1B 57 00	ESC W 0 -> Weit AUS
USR2:	06CE:	↑W	03 1B 57 01	ESC W 1 -> Weit EIN
USR3:	06D3:	↑E	01 0F	SI=CHR\$(15) -> Eng EIN
USR4:	06D8:	↑R	03 12 1B 50	DC + ESC P -> Eng AUS+NORMAL
RIBBON:	06DD:	↑Y	03 1B 53 00	ESC S 0 -> Hochstellen EIN
RIBOFF:	06E2:	↑Y	04 1B 54 1B 48	ESC T +H -> Hochstellen AUS
ROLUP:	06BF:	↑T	03 1B 53 01	ESC S 1 -> Tiefstellen EIN
ROLDOW:	06C4:	↑T	04 1B 54 1B 48	ESC T +H -> Tiefstellen AUS
PSINIT:	06E7:		07 1B 40 1B 52 02 1B 4F	-> dt. Drucker INIT
PSFINI:	06FB:		09 1B 57 00 12 1B 40 1B 52 02	-> Drucker END
BLDSTR:	0691:	↑B	03	Fettdruck 3mal Anschlag
DBLSTR:	0692:	↑D	02	Doppeldruck 2mal Anschlag

Tabelle 1 Patches für den mnemotechnischen WORDSTAR 3.0 mit den frei wählbaren deutschen oder ASCII-Sonderzeichen.

Man kommt zum Ziel, wenn diese Umschaltung mit einem Ctrl-Code ermöglicht wird. Hierzu bietet Wordstar die Labels **PALT:** und **PSTD:** an. Diese haben die Adressen 06B5H bzw. 06BAH und werden über ↑A und ↑N aktiviert. In der Praxis verfährt man wie folgt: Der beim Booten angebotene Sprachsatz ist zunächst deutsch. Soll nun im Verlauf des Editierens ein ASCII-Zeichen ausgedruckt werden, so gibt man ↑A (= Ctrl-P-Ctrl-A) und drückt dann diejenige Taste, die dem ASCII-Zeichen entspricht (siehe **Tabelle 2**). Danach schaltet man wieder mittels ↑N (= Ctrl-P-Ctrl-N) auf den deutschen Satz zurück. ↑A ist also der (A)lternative, ↑N der (N)ormal installierte Satz. Wichtig ist, daß die Drucker-Initialisierung deutsch erfolgt. Diese Festlegung erfolgt im Label **PSINIT:** in Adresse 06E7H (WS V3.0) und wird in **PSFINI:** mit 1B 52 02

wiederholt. So wird dafür gesorgt, daß der Neuausdruck einer Datei – trotz eines ver-gessenen ↑N – mit keinem anderen als dem deutschen Zeichensatz startet. Die Vorgehensweise des Patchens wurde bereits in dem vorhergehenden Aufsatz erläutert. In der nachstehenden **Tabelle 1** wurden die übrigen Patches für die unterschiedlichen Schriftarten (breit, eng, gesperrt, fett, doppelt, indiziert) so gewählt, daß die Ctrl-Codes mnemotechnische Regeln befolgen. Alle wichtigen Labels sind aufgeführt. Zu beachten ist, daß das ↑-Zeichen für „Ctrl-P“ steht. **Tabelle 3** nennt die Label-Inhalte für die neun verschiedenen Zeichensätze, die beispielsweise der FX-80 anbietet. Man kann sich so eine deutsch-französische oder deutsch-spanische WS-Version anlegen und würde, wollte man nur in französischer Sprache schreiben, gleich zu Be-

ginn der Datei ein ↑A setzen. Dieses „FLAG“ müßte nicht einmal durch ↑N zurückgesetzt werden, da bei Druckbeendigung der Inhalt von PSFINI: automatisch wieder für deutsche Verhältnisse sorgt. Die Texte der **Abbildungen 1-3** zeigen zweisprachige Beispiele.



**Sonderzeichen mit Wordstar**  
 Technisch wissenschaftliche Texte enthalten in der Regel viele Sonderzeichen, beispielsweise die Elemente des griechischen Alphabets oder die Buchstaben individueller Schriften, wie inverse oder gotische Zeichen. Auch sind ganze Sequenzen zusammengesetzter Grafikelemente denkbar, mit denen Bordüren oder besondere Hervorhebungen des Textes erzeugt werden.  
 Derartige „Fonts“ können in den Puffer der Epson-Drucker der FX-Serie geladen werden und stehen zusätzlich zum übrigen Zeichensatz zur Verfügung. Die Fonts bezieht man entweder aus bekannten Programmen, oder man erstellt sie sich mit Hilfe eines Shape-Editors und einem Font-Loader nach eigener Vorstellung und Geschmack. Die letztgenannte Möglichkeit ist oftmals die bessere, da man nur diejenigen Sonderzeichen auswählt und auf die Speicherplätze des Puffers verteilt, die wirklich auch benötigt werden. Dabei ist es zweckmäßig, mnemotechnischen Regeln folgend, ein Alpha auf die Taste A, ein Beta auf B und das Omega-Zeichen auf O zu legen.  
 Zur Übernahme eines Sonderzeichens aus dem Puffer erwartet der FX-80 den Befehl ESC % 1 0, während die Rückerschaltung auf den normalen Zeichensatz durch ESC % 0 0 bewirkt wird.

PALT: 06B5 ↑A 03 1B 52 00	= ASCII / USA EIN
	01 = Frankreich
	02 = Deutschland
	03 = England
EIN:	04 = Dänemark
Ctrl-P-Ctrl-A	05 = Schweden
	06 = Italien
	07 = Spanien
	08 = Japan
PSTD: 06BA ↑N 03 1B 52 02	= Fremdsprachen AUS
AUS:	
Ctrl-P-Ctrl-N	

**Tabelle 3** LABEL-Inhalte für die einzelnen Fremdsprachen-Sonderzeichen mit Deutsch als Default. Pro WORDSTAR-Diskette kann immer Deutsch mit einer ausgewählten Fremdsprache kombiniert werden.

Zur Aufnahme dieses Befehlsatzes in Wordstar bieten sich – falls nicht anderweitig vergeben – die Labels PALT: und PSTD: mit dem Aufruf ↑A (alternativer Satz) bzw. ↑N (normaler Satz) an. Die Label-Inhalte werden mit dem INSTALL-Programm wie folgt gesetzt:

```
PALT: 04 1B 25 01 00
PSTD: 04 1B 25 00 00
```

Ein weiterer Schritt muß nun folgen; denn es lauert eine böse Falle bei der Drucker-Initialisierung PSINIT: und dem Druck-Ende PSFINI:. Dort wird normalerweise mittels 1B 40 = „alles auf normal zurücksetzen“ der Puffer gelöscht. Daher muß dieser Befehl ersetzt werden durch eine Reihe von Einzelbefehlen, die das Löschen des Puffers nicht bewirken. Die genannten Labels sind 16 Bytes lang und werden wie folgt (gleich) belegt:

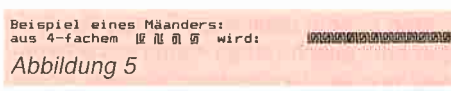
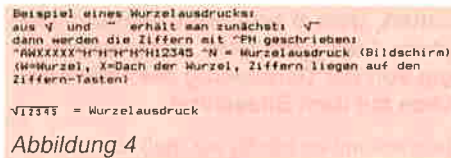
```
PSINIT: 10 1B 54 1B 25 00 00 1B 4F 1B
57 00 12 1B 46 1B 48
PSFINI: 10 1B 54 1B 25 00 00 1B 4F 1B
57 00 12 1B 46 1B 48
```

Damit ist die Patch-Arbeit beendet. Im Text schaltet man die Font-Zeichen mittels Ctrl-P-A (= ↑A) ein und mit Ctrl-P-N (= ↑N) wieder aus und damit auf den normalen Zeichensatz zurück.

Die folgende Bildschirm-Textzeile – die ↑AA ↑N-Messung liefert 20 ↑AO ↑N nach dem ↑AAB ↑N-Verfahren – ergibt ausgedruckt: – die α-Messung liefert 20 α nach dem αP-Verfahren –  
 Zeichen können beliebig zusammengesetzt werden. So entsteht eine Wurzelzahl

(**Abbildung 4**) durch einen Wurzelzeichen-Font mit seinen „Dach-Verlängerungen“ und der – mittels dem ↑PH-Backspace – darunter gesetzten Ziffern. Letztere sind wiederum Font-Elemente, da sie kleiner sein müssen, um unter das Wurzelzeichen zu passen.

Mit nur 4 Grundgrafik-Elementen läßt sich das in **Abbildung 5** dargestellte Mäanderband erzeugen, wobei die Elemente in Mehrfach-Serien aufgerufen werden.



**Tabelle 4** zeigt eine typische Font-„Komposition“. Die relevante Tastenzuordnung kann mit Hilfe des nachstehenden Applesoft-Programms ausgedruckt werden.  
 100 D\$ = CHR\$(4) : HOME  
 110 E\$ = CHR\$(27) + CHR\$(37) + CHR\$(1) + CHR\$(0)  
 120 A\$ = CHR\$(27) + CHR\$(37) + CHR\$(0) + CHR\$(0)  
 130 PRINT : PRINT D\$“PR#0”  
 140 PRINT D\$“PR#1” : PRINT : PRINT  
 150 FOR I = 32 TO 127  
 160 PRINT CHR\$(I); “ ”; E\$; CHR\$(I); A\$  
 170 NEXT  
 180 PRINT D\$“PR#0”  
 Nachzutragen ist, daß alle Fonts den entsprechenden Normalschriften Breit, Eng, Sperrschrift, Italic, Elite, hoch- und tiefgestellt, Fett und Doppelt sowie Proportional-schrift folgen, so daß hierbei eine ungeahnte Fülle von Varianten zur Verfügung stehen.



**Blick über den Zaun:  
Die moderne Satztechnik**

Da sich die Terminologie der Matrixdrucker-Hersteller nicht mit der des grafischen Gewerbes deckt, wollen wir auch hier noch einmal einen Blick über den Zaun werfen. In der Drucktechnik unterscheidet man zwischen Schriftgruppe, Schriftart, Schriftschnitt und Schriftgrad. Die *Schriftgruppe* (group of typeface) ist eine von 10 lateinischen Schriftkategorien („französische Renaissance-Antiqua“ usw.). Die *Schriftart* (typeface) ist eine individuelle Schöpfung einer Schrift (z.B. „Garamond“), die sich einer Schriftgruppe subsumieren läßt. Es gibt ca. 2000 Schriftarten. Der *Schriftschnitt* (typecut; auch font genannt) ist eine von mehreren Erscheinungsformen *derselben* Schriftart (z.B. fett, halbfett, mager, kursiv usw.; Sperrschrift gilt dabei nicht als gesonderter Schriftschnitt). Der *Schriftgrad* (type size) ist die in Punkt gemessene Größe eines Schriftschnittes (kontinentaler Punkt

□	□	□	□	→	←	{	}		±	≡	•	//	1	2	3	4	5	6	7	8	9	0	□	□	≠	∞	∞	
→	←	↖	↗	↘	↙	↕		∧	∩	∪	∩	∪	∩	∪	∩	∪	∩	∪	∩	∪	∩	∪	∩	∪	∩	∪	∩	∪
↑	α	β	γ	δ	ε	ζ	η	θ	ι	κ	λ	μ	ν	ξ	ο	π	ρ	σ	τ	ω	ϕ	ψ	χ	ψ	χ	ψ	χ	ψ
□	□	□	□	→	←	{	}		±	≡	•	//	1	2	3	4	5	6	7	8	9	0	□	□	≠	∞	∞	
→	←	↖	↗	↘	↙	↕		∧	∩	∪	∩	∪	∩	∪	∩	∪	∩	∪	∩	∪	∩	∪	∩	∪	∩	∪	∩	∪
↑	α	β	γ	δ	ε	ζ	η	θ	ι	κ	λ	μ	ν	ξ	ο	π	ρ	σ	τ	ω	ϕ	ψ	χ	ψ	χ	ψ	χ	ψ
□	□	□	□	→	←	{	}		±	≡	•	//	1	2	3	4	5	6	7	8	9	0	□	□	≠	∞	∞	
→	←	↖	↗	↘	↙	↕		∧	∩	∪	∩	∪	∩	∪	∩	∪	∩	∪	∩	∪	∩	∪	∩	∪	∩	∪	∩	∪
↑	α	β	γ	δ	ε	ζ	η	θ	ι	κ	λ	μ	ν	ξ	ο	π	ρ	σ	τ	ω	ϕ	ψ	χ	ψ	χ	ψ	χ	ψ

**Tabelle 4**

0,375mm, angloamerikanischer Punkt 0,351mm). Die Summe aller Schriftschnitte heißt *Schriftfamilie* (type family); die Summe aller Schriftzeichen eines Schriftschnittes (beim veralteten Bleisatz in allen Schriftgraden) heißt *Schriftgarnitur* (type font). U. Stiehl



# Wollen Sie mit Ihrem MACINTOSH grafisch arbeiten?



**Das MacTablett von Summagraphics – mit entsprechender Software – macht es möglich.**

**Informieren Sie sich bei Ihrem Apple Händler, oder direkt unter Telefon 089/1415077**



**Summagraphics**®  
LIMITED

Niederlassung Deutschland

Georg-Brauchle-Ring 68  
D-8000 München 50  
Telefon 089/1415077  
Telex 5214793

# Tips und Tricks in Pascal

## Teil 1: Die versteckte Prozedur „ldsearch“

oder wie man Schlüsselwörter halbfett druckt

von Dieter Geiß

In dieser mehrere Teile umfassenden Serie soll über viele interessante Themen für den an Pascal interessierten Peeker-Leser berichtet werden. Diese werden vor allem programmertechnische Kniffe und Eigenheiten des UCSD-Systems sein, die nicht in den Handbüchern stehen und die man nur herausfinden kann, wenn man sich lange und intensiv mit dem System beschäftigt.

### Das Problem

Es sei folgende Aufgabe gestellt: Ein vorhandener Pascal-Quelltext, also z.B. der SYSTEM.WRK.TEXT, soll auf den Drucker oder auch ein anderes Gerät ausgegeben werden. Dabei sollen die Pascal-Schlüsselwörter (IF, THEN, ELSE, FOR, DO, WHILE...), also die vom Compiler reservierten Bezeichner (Reserved Words), besonders hervorgehoben werden, z.B. fett oder auch unterstrichen, wie dies auch in der Literatur zum Teil üblich ist. Das Programm soll möglichst kurz sein, dabei aber auch Kommentare und konstante Zeichenketten erkennen können, in denen die reservierten Bezeichner nicht hervorgehoben werden dürfen.

### Die Theorie

Die Lösung ist eigentlich ganz einfach – der Compiler muß die reservierten Wörter schließlich auch erkennen können. Warum also nicht die (schnelle) Prozedur benutzen, die auch vom Compiler eingesetzt wird?

Es handelt sich hierbei um „ldsearch“, die eine UCSD-Standardprozedur ist. Das Problem ist nur, daß sie nirgends erklärt wird und man deswegen nicht weiß, mit welchen Parametern sie aufgerufen wird und was sie genau bewirkt. Um das herauszufinden, muß man entweder ein Compiler-Source-Listing haben oder die Prozedur im Interpreter genau untersuchen. Da ein Source-Listing im allgemeinen nicht vorliegt, muß man auf die zweite Methode zurückgreifen.

Zunächst aber eine kurze Vorbemerkung: Ein **Bezeichner** (Identifier) ist eine Folge von Zeichen, bestehend aus mindestens einem Buchstaben (A–Z), gefolgt von beliebig vielen Buchstaben oder Ziffern. Nur die ersten acht Zeichen sind jedoch in der Apple-Version des UCSD-Compilers signifikant. Dies gilt sowohl für die Version 1.1 als auch für 1.2.

Findet man in einer Zeichenkette also einen Buchstaben, so weiß man, daß es sich um einen Bezeichner handeln muß, der so lang ist, bis ein Trennzeichen folgt, also ein (ASCII-)Zeichen, welches kein Buchstabe und keine Ziffer ist.

An diesem Punkt setzt die Prozedur „ldsearch“ an. An diese wird ein Zeiger (Cursor) sowie ein String oder ein Packed Array of Char übergeben. Der Zeiger muß auf den ersten Buchstaben des Wortes zeigen, das man untersuchen möchte. In **Tabelle 1** ist ein Beispiel wiedergegeben. Dabei soll 14 die Länge des Strings sein, die sich im ersten Byte (Byte 0 des

Strings) befindet. Symcursor hätte also den Wert 1, da das „w“ von while der erste Buchstabe des Strings ist.

```
0 12345678901234
S [14]: while I < 5 do
      ↑
      Symcursor
```

Tabelle 1

Nun ruft man ldsearch (Symcursor, S) auf. In der Prozedur geschieht nun folgendes: Es wird untersucht, ob ein Pascal-Schlüsselwort vorliegt. Dabei wird Symcursor auf das Ende des zu untersuchenden Wortes gestellt (siehe **Tabelle 2**).

```
0 12345678901234
S [14]: while I < 5 do
      ↑
      Symcursor
```

Tabelle 2

Gleichzeitig werden aber auch noch drei weitere Variablen beeinflusst, die nicht am Aufruf beteiligt sind, aber in der Variablendefinition direkt hinter der Variablen „Symcursor“ definiert sein müssen (siehe Listing).

– In „Sym“ wird die Nummer des Symbols abgespeichert, das gerade gefunden wurde, wenn es sich um ein Schlüsselwort handelt. Der Einfachheit halber wurde „Sym“ als Integer-Zahl definiert, was für diesen Zweck ausreicht. Eigentlich müßte „Sym“ eine Variable vom Aufzählungstyp sein, also Sym: (Ident, Comma, Colon, ..., DOsy, TOsy,...). Ist Sym = 0, so handelt

es sich um kein reserviertes Wort, also um einen Bezeichner. Ist Sym < > 0, so liegt ein reserviertes Schlüsselwort vor. Für while ist die Nummer z.B. 23.

– Die nächste Variable „Op“ wird gesetzt, wenn es sich um einen Operator handelt, also z.B. AND, DIV, MOD, OR, IN. Wird ein solcher gefunden, ist Op < > 0 und Sym 39, 40 oder 41. „Op“ spielt für unser Problem aber keine Rolle.

– Die Variable „Id“ wird ebenfalls beeinflusst. „Id“ ist vom Typ „Alpha“ und kann einen Bezeichner (maximal acht Buchstaben) enthalten. Wenn Sym = 0 ist, dann muß es sich um einen selbst-definierten Bezeichner oder um eine vordefinierte Prozedur wie „Unitread“ etc. handeln. „Id“ enthält dann nach dem Aufruf von „Idsearch“ die ersten acht Buchstaben des gefundenen Bezeichners, die sogar schon in Großbuchstaben umgewandelt sind. Man könnte „Id“ also dazu benutzen, um etwa mit der Funktion „tree-search“ (siehe Pascal Language Reference Manual, S. 49) den Bezeichner weiter zu untersuchen oder in einen Binärbaum einzufügen, um später ein sog. Cross-Reference-Listing von allen Bezeichnern zu bekommen.

## Das Programm

Da das Programm möglichst kurz sein sollte, wurde auf schnelle Ein/Ausgabe-Routinen verzichtet und statt dessen die eingebauten Prozeduren „Readln“ und „Writeln“ benutzt, die allerdings bei Textfiles nicht besonders schnell laufen. Wer möchte, kann an dieser Stelle noch Verbesserungen anbringen.

Die Abfrage nach Kommentaren und konstanten Zeichenketten ist komplizierter als die Bezeichnersuche selbst, die sich in wenigen Zeilen abhandeln läßt.

Nach dem Starten des Programms werden zuerst Ein- und Ausgabe-Files erfragt. Läßt sich ein File nicht öffnen, wird das Programm verlassen. Danach kann man einen Präfix- und einen Suffix-String eingeben. Diese Strings werden dem gefundenen Pascal-Schlüsselwort voran- oder nachgestellt. Will man beispielsweise die Schlüsselwörter unterstrichen ausgeben, so gibt man für den Epson-Drucker als Präfix die Zeichenfolge <ESC> ,-' ,1' und als Suffix <ESC> ,-' ,0' ein, für den Imagewriter <ESC> ,X' und <ESC> ,Y'. Schließlich kann man noch wählen, ob man eine Zeilennummerierung haben möchte.

## IDSEARCH

```
{%C von Dieter Geiß, 28-November-1984}
{$R-}

program FastIdsearch (input, output);

type Alpha    = packed array [0..7] of char;
   MaxStr     = string [255];

var Lines      : integer;
   Lnum       : boolean;
   Comment     : integer;
   Comments    : array [1..4] of string [2];
   S          : MaxStr;
   Pre        : string;
   Post       : string;
   Infile     : interactive;
   Outfile    : interactive;

(-----)

procedure Init;

var C : char;

begin {Init}
  {$I-}
  page (output);
  Lines := 0;
  writeln ('Schnelle Schlüsselwortsuche mit idsearch');
  writeln ('von Dieter Geiß');
  writeln;
  write ('Eingabefile: ');
  readln (S);
  openold (Infile, S);           {reset geht auch}
  if IOresult <> 0 then
  begin
    openold (Infile, concat (S, '.TEXT'));
    if IOresult <> 0 then exit (program)
  end; {if}
  write ('Ausgabefile: ');
  readln (S);
  if S = '' then S := 'CONSOLE: ';
  opennew (Outfile, S);         {rewrite geht auch}
  if IOresult <> 0 then exit (program);
  writeln;
  write (' Präfix-String: ');
  readln (Pre);
  write (' Postfix-String: ');
  readln (Post);
  writeln;
  write ('Zeilennummerierung? ');
  read (C);
  writeln;
  writeln;
  Lnum := C in ['J', 'j'];
  Comment := 0;
  Comments [1] := '{';
  Comments [2] := '{*';
  Comments [3] := '}';
  Comments [4] := '*}'
  {$I+}
end; {Init}

(-----)

procedure Scanner (var S : Maxstr);

var SymCursor : integer;
   Sym        : integer;
   Op         : integer;
   Id         : Alpha;
   OldCursor  : integer;
   P          : integer;

begin {Scanner}
  SymCursor := 1 + scan (length (S), <> ' ', S [1]);
  if Comment <> 0 then {Kommentar möglicherweise...}
  begin
    {...in dieser Zeile zu Ende}
    P := pos (Comments [Comment + 2], S);
    if P <> 0 then
    begin
      SymCursor := P + length (Comments [Comment + 2]);
      Comment := 0;
    end {if}
  end; {if}
end; {if}
```



Übrigens gibt es noch weitere Prozeduren, die der Compiler erkennt und nirgends beschrieben werden. Diese haben allerdings keine große Bedeutung. „Openold“ beispielsweise ist nämlich gleichbedeutend mit „Reset“, und „Opennew“ entspricht „Rewrite“.

„Time (I1, I2)“ füllt beim Apple beide Integer-Zahlen I1 und I2 mit 0, weil keine Uhr vorhanden ist. Hätte man eine Uhr abgeschlossen, müßte man den P-Code-Interpreter an der Stelle patchen, wo die Adresse der Standardprozedur „Time“ steht (\$D112 und \$D113 in der Bank 1). Übrigens hat „Segment“ den gleichen Id-Code wie „Program“. Man könnte also statt „Program Test“ auch „Segment Test“ schreiben, statt „Segment Procedure“ auch „Program Procedure“.

Im nächsten Teil soll der Frage nachgegangen werden: Kann man den P-Code optimieren?

```

while (SymCursor <= length (S)) and (Comment = 0) do
begin
  OldCursor := SymCursor;
  if S [SymCursor] in ['A'..'Z', 'a'..'z']
  then
    begin
      idsearch (SymCursor, S);
      if Sym <> 0 then
        begin
          insert (Post, S, SymCursor + 1);
          insert (Pre, S, OldCursor);
          SymCursor := SymCursor + length (Post) + length (Pre)
        end {if}
      end {if}
    else
      if S [SymCursor] = ' ' {konstante Zeichenkette}
      then SymCursor := SymCursor + 1 +
        scan (length (S) - SymCursor, = ' ', S [SymCursor + 1])
      else
        begin
          if S [SymCursor] = '{'
          then Comment := 1
          else if pos ('*', S) = SymCursor then Comment := 2;
          if Comment <> 0 then
            begin
              P := pos (Comments [Comment + 2], S);
              if P > SymCursor then {Ende des Kommentars,...}
                begin
                  {...in gleicher Zeile}
                  SymCursor := P + length (Comments [Comment + 2]) - 1;
                  Comment := 0
                end {if}
            end {if}
          end; {else, else}
          SymCursor := SymCursor + 1
        end {while}
      end; {Scanner}
    end;

    (-----)

begin {FastIdsearch}
  Init;
  while not eof (Infile) do
  begin
    fillchar (S, size_of (S), 0);
    readln (Infile, S);
    Scanner (S);
    Lines := Lines + 1;
    if Lnum then write (Outfile, Lines : 5, ' ');
    writeln (Outfile, S)
  end; {while}
  close (Infile);
  close (Outfile, lock)
end {FastIdsearch}.

```



## SUPERDUMP und Apple IIc

Das Programm Superdump aus Peeker 6/85, S. 22 läuft ohne Änderung nicht auf dem Apple IIc. Die Ursache dieses Problems liegt nicht in einer Unzulänglichkeit des Programms, sondern in der eigenwilligen Konstruktion der seriellen Karte im IIc (die Image-Writer-Toolkit-Diskette unter DOS 3.3 funktioniert ebenfalls nicht auf dem IIc.).

Bei der Übertragung eines Zeichens wird in einem Hardwareregister der Status des Druckers abgefragt. Ein gesetztes Bit 6 bei der Super Serial Card signalisiert die Bereitschaft des Druckers, ein neues Zeichen entgegenzunehmen. Dieses Register wurde beim IIc geändert, so daß nun das Bit 5 diese Aufgabe übernimmt.

Um das Programm auf den Apple IIc anzupassen, muß die Routine zur seriellen Ausgabe geändert werden. Diesen Patch kann man von der Apple-Version abhängig machen, um das Programm auf den verschiedenen Rechnern starten zu können.

Dazu muß folgende Zeile in das Applesoft-Programm SUPERDUMP.IMAGEWRITER eingefügt werden:

```
1325 IF PEEK (64435) = 6 AND PEEK (64448) = 0 THEN POKE 34499,32
```

Die beiden PEEK-Anweisungen fragen die sog. Machine-ID-Bytes ab, wodurch nur im Falle des IIc das entsprechende Byte in der Ausgaberroutine geändert wird.

# Assembler-Pseudo-Opcode-Referenztafel

von Dr. Jürgen B. Kehrel

Für den Apple II gibt es rund ein Dutzend verschiedener Assembler. In ihren Standardbefehlen halten sich fast alle an die Opcodes (= Operation Codes, Befehls- worte), die von dem ersten Hersteller des 6502-Prozessors, MOS Technology Inc., eingeführt wurden. Ein guter Assembler verfügt darüber hinaus noch über eine Anzahl von sogenannten Pseudo-Opcodes, die nicht zur Steuerung des 6502 dienen, sondern vielmehr Befehle für den Assembler selber darstellen. Diese sind nicht genormt, so daß eine direkte Übertragung auf einen anderen Assembler meistens nicht möglich ist, auch wenn viele Ähnlichkeiten vorliegen. Im *Peeker* sollen bevorzugt Listings im Big-Mac- oder Merlin-Format veröffentlicht werden, was die anderen Assembler natürlich nicht ausschließt.

Um Ihnen eine Hilfe zur Übertragung von fremden Pseudo-Opcodes auf Ihr eigenes System zu geben, sind nachfolgend die Befehls- worte der fünf gebräuchlichsten Assembler einander gegenübergestellt. Halbfett gedruckt finden Sie in alphabetischer Reihenfolge die Liste der Big-Mac- bzw. Lisa-2.5-Befehle und ihre Entsprechungen bei den übrigen Assemblern. Ist ein Code eingeklammert, dient er nur unter speziellen Bedingungen als Ersatz. Ein Strich in einer Rubrik bedeutet, daß der entsprechende Befehl nicht vorhanden ist. Das schließt aber nicht aus, daß Sie über einen Umweg nicht auch zum selben Ergebnis kommen können. Alle Angaben geschehen nach bestem Wissen, aber ohne Gewähr, denn ich arbeite natürlich nicht mit all diesen Assemblern.

Tabelle siehe nächste Seite

## ProDOS-Analyse

Version 1.0.1, 1.0.2, 1.1.1

Arne Schäpers

1985, ca. 450 S., kart.,  
DM 68,-  
ISBN 3-7785-1134-3

Dr. Alfred Hüthig Verlag  
6900 Heidelberg · Postfach 10 28 60

„Die ProDOS Analyse“ ist die umfangreichste und detaillierteste Darstellung, die jemals ein Apple-Betriebssystem erfahren hat. Wer die „Innereien“ von ProDOS bis zum letzten Byte, z. T. bis ins letzte Bit kennenlernen möchte, braucht dieses Buch. Das komplette Betriebssystem (Urlader, MLI, Disk-Driver, RAM-Disk-Driver und Uhr-Routine) mit Ausnahme des BASIC-SYSTEM wird mit umfangreichen Kommentaren und Übersichtstabellen disassembliert. Dabei werden alle bisherigen Versionen von 1.0.1 bis 1.1.1 berücksichtigt. „Die ProDOS Analyse“ beschreibt erstmals auch mehrere Programmierfehler, die bis

in die neueste Version zu finden sind. Auch die nicht im „Technical Reference Manual“ aufgeführten Eigenschaften von ProDOS werden analysiert und beschrieben, z. B. die vertrackten eingebauten Testroutinen zur Identifikation der verschiedenen Apple II Modelle und eventueller Nachbaugeräte. Programmierer, die ProDOS versionsabhängig „patches“ möchten, erhalten hier den genauen Überblick, wo was geändert werden muß, damit dies keine negativen Konsequenzen hat. Durch die minutiöse theoretische Sezierung von ProDOS eröffnen sich völlig neue programmierpraktische Perspektiven.

Big Mac	Merlin	Lisa 2.5	Toolkit	S-C Macro	Anmerkung
DA	DA	<b>ADR</b>	DW	.DA	2-Byte-Adresse / Ausdruck Lo-Hi
<b>ASC</b>	ASC	ASC	ASC	.AS	ASCII-String (" = Bit 7 gesetzt)
<b>AST</b>	AST	-	(REP/CHR)	-	Drucke Anzahl Sternchen
FLS	FLS	<b>BLK</b>	-	-	FLASH-ASCII-String
<b>BGE</b>	BGE	BGE	BGE	BGE	= BCS
<b>BLT</b>	BLT	BLT	BLT	BLT	= BCC
DA#	DA#	<b>BYT</b>	DB	.DA#	Adresse / Ausdruck nur Lo-Byte
<b>CHK</b>	CHK	-	-	-	Kontrollzahl (Checksum)
<b>DA</b>	DA	ADR	DW	.DA	2-Byte-Adresse / Ausdruck Lo-Hi
<b>DDB</b>	DDB	<b>DBY</b>	DDB	-	2-Byte-Adresse / Ausdruck Hi-Lo
<b>DCI</b>	DCI	DCI	(DCI)	.AT	ASCII-String, letztes Bit invertiert
-	-	<b>DCM</b>	-	.TF	Führe DOS-Befehl aus
<b>DDB</b>	DDB	DBY	DDB	-	2-Byte-Adresse / Ausdruck Hi-Lo
<b>DFB</b>	DFB	(HBY/BYT)	DFB	.DA#	Definiere Byte
(DS)	(DS)	<b>DFS</b>	(DS)	(.BS)	Reserviere Speicher
<b>ELSE</b>	ELSE	.EL	ELSE	.EL	Bedingte Assemblierung ELSE
<b>END</b>	END	END	END	.EN	Programmende
<b>EOM/&lt;&lt;&lt;&lt;</b>	EOM/<<<<	-	-	.EM	Ende einer Macrodefinition
EQU	EQU	<b>EPZ</b>	EQU	.EQ	Lisa: Zero-Page-Label 1 Byte
<b>EQU</b>	EQU	EQU	EQU	.EQ	Label 1 oder 2 Byte (Lisa 2 Bytes)
<b>EXP ON</b>	EXP ON	-	-	.LIST	Drucke Macros aus
<b>FIN</b>	FIN	<b>.FI</b>	FIN	.FI	Bedingte Assemblierung Ende
<b>FLS</b>	FLS	BLK	-	-	FLASH-ASCII-String
TR OFF	TR OFF	<b>GEN</b>	-	-	Liste alle Bytes pro Zeile
DA#>	DA#>	<b>HBY</b>	DW#>	.DA/	Adresse / Ausdruck nur Hi-Byte
<b>HEX</b>	HEX	HEX	HEX	.HS	1-Byte Hexadezimalwert
PUT	PUT	<b>ICL</b>	CHN	.IN	Verkette Quellcodefiles
DO	DO	<b>.IF</b>	DO	.DO	Bedingte Assemblierung Anfang
<b>INV</b>	INV	INV	-	-	INVERSE-ASCII-String
-	-	<b>LET</b>	-	-	Label Neudefinition
<b>KBD</b>	KBD	-	-	-	Labeleingabe über Tastatur
<b>LST ON</b>	LST ON	LST	LST ON	.LIST ON	Listing anschalten
<b>MAC</b>	MAC	-	-	.MA	Beginn Macrodefinition
LST OFF	LST OFF	<b>NLS</b>	LST OFF	.LIST OFF	Listing ausschalten
TR ON	TR ON	<b>NOG</b>	-	-	Listet nur 3 Bytes pro Zeile
<b>OBJ</b>	OBJ	OBJ	OBJ	OBJ	Aktuelle Objekt-Code Adresse
<b>ORG</b>	ORG	ORG	ORG	ORG	Adresse für ablauffähigen Code
<b>PAG</b>	PAG	PAG	PAGE	.PG	Sende Ctrl-L (neue Seite)
<b>PAU</b>	PAU	PAU	-	-	Pause, Abbruch
-	-	<b>PHS</b>	-	-	Neuer ORG, ohne OBJ zu ändern
<b>PCM/&gt;&gt;&gt;&gt;</b>	PCM/>>>>	-	-	unnötig	Macroaufruf
<b>PUT</b>	PUT	ICL	CHN	.IN	Verkettung von Quellcode-Files
<b>SAV</b>	SAV	(DCM)	-	.TF	Automat. Obj.-Code Abspeicherung
<b>SKP</b>	SKP	-	-	-	Sende Zeilenvorschub
-	-	<b>STR</b>	-	-	ASCII-String mit Längenbyte
<b>TR ON</b>	TR ON	NOG	-	-	Liste nur 3 Bytes pro Zeile
-	-	<b>TTL</b>	SBTL	.TI	Titel auf jeder Ausdruckseite
<b>VAR</b>	USR	<b>USR</b>	-	.US	Benutzerdefinierbarer Befehl
-	VAR	-	-	-	Labeldefinition in Macros
-	<b>ERR</b>	-	-	-	Fehlermeldung, wenn Ausdruck <> 0
-	<b>LUP</b>	-	-	-	Schleifenanfang
-	-↑	-	-	-	Schleifenende
-	<b>REV</b>	-	-	-	Wie ASC, nur String rückwärts
-	-	-	<b>CHR</b>	-	Definiere Byte für REP
-	-	-	<b>DEND</b>	-	Dummyteil Ende
-	-	-	<b>SECT</b>	-	Dummyteil Anfang
-	-	-	<b>ENTRY</b>	-	Globales Label (unbenutzt)
-	-	-	<b>EXTRN</b>	-	Externes Label (unbenutzt)
-	-	-	<b>MSB ON</b>	-	Setzt Bit 7 in Strings
-	-	-	<b>MSB OFF</b>	-	Löscht Bit 7 in Strings
-	-	-	<b>REL</b>	-	Verschiebbarer Code für RLOAD
-	-	-	<b>REP</b>	-	Druckt CHR mehrmals
#Adresse	#Adresse	#Adresse	#Adresse	#Adresse	Lo-Byte eines 2-Byte-Ausdrucks
#<Adresse	#<Adresse	-	#>Adresse!	#<Adresse	Lo-Byte eines 2-Byte-Ausdrucks
#>Adresse	#>Adresse	-	#<Adresse!	#>Adresse	Hi-Byte eines 2-Byte-Ausdrucks
#/Adresse	#/Adresse	/Adresse	-	#/Adresse	Hi-Byte eines 2-Byte-Ausdrucks







## peeker-Börse

Vorname, Name

Beruf

Straße

Wohnort

PLZ

Bitte veröffentlichen Sie den umstehenden Text von \_\_\_\_\_ Zeilen à \_\_\_\_\_ DM in der nächsterreichbaren Ausgabe von »peeker«

**Bei Angeboten:** Ich bestätige, daß ich alle Rechte an den angebotenen Sachen besitze

Datum

Unterschrift



## Produkt-Karte

Karte bitte vollständig ausfüllen

Vorname, Name

Firma

Straße

PLZ/Ort

Telefon mit Vorwahl

Anschrift der Firma angeben, bei der Sie bestellen bzw. von der Sie Informationen wünschen



## Info-Karte

Karte bitte vollständig ausfüllen

Vorname, Name

Firma

Straße

PLZ/Ort

Telefon mit Vorwahl

## ANTWORTKARTE

### peeker-Börse

Anzeigen-Service

Dr. Alfred Hüthig Verlag

Postfach 10 28 69

6900 Heidelberg 1

## POSTKARTE

Firma

Straße

PLZ/Ort

## POSTKARTE

### peeker

Redaktion

Postfach 10 28 69

6900 Heidelberg 1



# Produkt-Karte

Wünschen Sie weitere Informationen zu einem der im Heft vorgestellten Produkte ?

Nichts einfacher als das. Produkt-Karte ausfüllen, mit 60-Pfennig frankieren und absenden.

Vorher aber nicht vergessen : kreuzen Sie an, welchen Informationswunsch Sie haben.

Damit erleichtern Sie dem Hersteller eine gezielte Beantwortung Ihrer Anfrage

Zum Schluß tragen Sie auf der Rückseite die genaue Anschrift des Inserenten/Herstellers und Ihre vollständige Firmenanschrift ein.

**PEEKER**  
MAGAZIN FÜR APPLE-COMPUTER

## Verkauf Hardware

**Apple-DOT-Matrix-Drucker** mit Parallel-Interface-Karte (Apple) VHB DM 1600,-  
H. Bösing, Erlenweg 8, 2872 Hude 2, Tel. 0 44 84/746

**APPLE comp. (6 Mon. Garantie.** Neugerät), mit 64 K, 6502 u. 280 CPU im IBM-Gehäuse, separater Tastatur, 80 Zeichen Disk-contr., 1 Laufw. 160 K, DM 1950,-  
Monitor bernst. DM 280,-  
Tel. 0 68 97/69 03

**APPLE-Komp. + Cards** zu Superpreisen! Ab DM 65,00! Liste von: RTE Software/TREIBER GmbH 7546 Enzklosterle

**Verk. APPLE IIc + Zub.** 094 09/933

**Org. APPLE Disk-Driver**, 2 Stück mit Controller 900 DM. PREH-Commander Tastatur 150 DM. Tel. 02 41/836 61

Neuwertig (3 Wochen im Einsatz) zu verkaufen: **2 corvus harddisc** à 6 MB sowie **1 disc-server**. Rückfragen bitte an Fa. Vitra, Hr. Dr. Treffert, Tel. 0 76 21/70 23 11

**Verkaufe TEAC-Doppellaufwerk** 2 x 640 KB mit R-PHI Controller. Beide ungebraucht VB: 1650,- DM, Tel. 09 31/41 32 58

**Fernschreiberinterface** am Gameport m. Programm DM 79,-  
P. Benner, Hubertusstr. 131, 4150 Krefeld

**BROSE 120 (csc)-Tastatur / 22 MHz Monitor (NEC)** Sonderpreise !!! Rüter, Hille 0 57 03/672

## Verkauf Software

**APPLE II: 'GIANT WORLD'** ist da!! \* Das neu Top-Adventure mit \*  
\* Spitzengrafik für riesigen \*  
\* Spielspaß! Es lohnt sich! \*  
Für nur 89,- DM Vorkasse o.NN bei \*\*\* FANTASTIC-Software \*\*\*  
\*\* Grasweg 7, 2857 Langen 3 \*\*

--- **STOCKMASTER II** ---  
Das Apple-Programm für echte **Börsengewinne**. Nur 485,- DM, Töngi, Computer-Praxis, Aspeltstr. 4, D-6500 Mainz 1.

**APPLE II emuliert SHARP-POCKETC.** Dat./Prg. ü. Game/11 Pin-C. Inf. geg. 2 x 0,80 DM in Bfm. Klaus Schmidt, Hasencleverstr. 25, 2000 Hamburg 74

**fig-FORTH --- DM 15,00**  
Assemblerlisting f. APPLE DM 15,-, FORTH auf Disk f. APPLE DM 45,-  
Info kostenlos --- C. Schmidt, Rungestr. 8, 3500 Kassel

**Verkaufe für Apple IIe/IIc** Originalprogramm 'Applevorks', Neueste Version (1.2.), VB 500 DM, Tel. 02 21/77 93 29.

## Ankauf Software

**Für II-e (II-c)** Programm gesucht das Daten auf Diskette bearbeitet  
Tel. 05 61/49 83 43 od. /804 48 38

**IIe-Neuling sucht Programme** aller Art. Liste bitte an Chiffre P 1003

## Verschiedenes

**APPLE REPARATUREN** (auch compatible M-boards, z.B. **Atlas, Arca, CES, Datastar, Dipa, Lasar, Mewa, PC-48+46, Plato, Radix, o. ae. sowie** Zusatzkarten und Disk-Drives) führt unser Spezialteam mit mehr als 5-jähriger Kunden- und Reparatur-Dienst-Erfahrung, garantiert zuverlässig und besonders kostengünstig aus. Bitte genaue Fehlerangabe sowie Tel.-Nr. für evtl. Rückfragen nicht vergessen.  
Auf Wunsch Kostenvoranschlag.  
**aaa-electronic gmbh**  
Habsburgerstr. 134, 7800 Freiburg, Tel. 07 61/27 68 64, Tx. 772 642 aaa d

\*\*\*\*\*  
\* **Floppysubsystem für Apple II** \*  
\* 2 x 3,5" TEAC FD-35F \*  
\* max. 2 x 1MB \*  
\* Autopatch-Controller \*  
\* inkl. Manual und SW \*  
\* ALU-Profil-Gehäuse m. Netz. \*  
\* komplett anschlussfertig \*  
\* nur DM 2398,- \*  
\*\*\*\*\*  
\* **Telekommunikation** \*  
\* Anschlusskabel \*  
\* Dataphon S21d \*  
\* Terminalsoftware \*  
\* komplett anschlussfertig \*  
\* nur DM 398,- \*  
\*\*\*\*\*  
\* **TISCH & ZETTL GdBR** \*  
\* **Elektronicvertrieb** \*  
\* **Rosenstr. 33, 8034 Germering** \*  
\* **Telefon 089/841 68 17** \*  
\*\*\*\*\*

**MC 3470** (der Leseverstärker auf dem Analogboard): DM 10,-, A. Deckers, PF 967, 7 Stuttgart 1

**Erscheinungstermin für Ausgabe 9/85 ist am 26. 8. 1985**

**Epson Interfaceumbau** f. AWorks DM 20. Mahr, Waldacker 71, 7300 Esslingen

Lieber **'ROBO'** bitte melden. Manfr. Rost, Düsseldorf, Suitbertusstr. 99

## Einkaufsführer



Keithstr. 26 · 1 Berlin 30 · ☎ 0 30-26 111 26



Bachstr. 104 · 2 HH 76 · ☎ 0 40-220 11 55

Für weitere Informationen zu einem der in dieser Ausgabe vorgestellten Produkte stehen Ihnen die Produktkarten zur Verfügung

Bitte verwenden Sie für Kleinanzeigen die vorgedruckten Antwortkarten in diesem Heft.



# Fakultäten

von Roland und Manfred Fietkau

„Sie nimpt viel Kopfs“  
Adam Ries (1492-1559)

In der Kombinatorik und Wahrscheinlichkeitsrechnung spielen Fakultäten eine wichtige Rolle. Der Ausdruck  $N!$  ( $N$ -Fakultät) bedeutet:  $1 \cdot 2 \cdot 3 \dots \cdot (N-1) \cdot N$ .  $N$  ist dabei eine positive natürliche Zahl. Zusätzlich muß definiert werden:  $0! = 1$ .

Fakultäten lassen sich im Prinzip sehr leicht berechnen; bereits eine einfache Applesoft-Zeile genügt:

```
10 F = 1: INPUT "Fakultät von ";N:
FOR I = 1 TO N: F = F * I: NEXT:
PRINT "F= ";F
```

Schon nach einigen Versuchen wird man jedoch an die Grenzen dieses einfachen Verfahrens stoßen. Zwar werden die Werte von  $0!$  bis  $12!$  noch exakt berechnet, doch ab  $13!$  liefert Applesoft nur noch Näherungswerte. Der Grund liegt darin, daß Mantisse und Exponent einer FP-Variablen nur 32 Bits umfassen. Konsequenterweise bricht das Applesoft-Programm auch ab  $34!$  mit einem Overflow-Error ab. Um auch höhere Fakultäten stellengenau berechnen zu können, muß man sich also einer anderen Methode bedienen. Mit (viel!) Papier und Bleistift ist die Berechnung leicht möglich: Man denke daran, daß vor der Entwicklung des Chips ganze Tabellen „zu Fuß“ berechnet wurden. Genau das ist die Methode, die auch das Assembler-Programm **FAKULTAET** benutzt.

## Das Verfahren

Das Programm wertet den numerischen Ausdruck nach „&“ aus und überträgt ihn nach  $\$0000$  und  $\$0001$ .

Um zeitaufwendiges Umrechnen (hex in dezimal) zu vermeiden, werden alle Berechnungen im Dezimalmodus (BCD-

Arithmetik) durchgeführt. Das bedeutet, daß der zu berechnende Ausdruck nicht größer als 9999 sein darf (9999! ist bereits eine Zahl mit über 35 000 Stellen).

Die zu berechnende Fakultät wird zunächst auf 1 initialisiert und dann fortlaufend mit den Zahlen von 1 bis  $N$  multipliziert. Da eine Multiplikationsroutine jedoch sehr zeitaufwendig ist, legt sich das Programm durch Aufaddieren eine Tabelle an, in der alle Multiplikationsergebnisse von  $0 \cdot N$  bis  $99 \cdot N$  eingetragen werden. Eine einfache Überlegung zeigt, daß dafür höchstens 3 Bytes pro Tabellenelement benötigt werden. Die aktuelle Doppelziffer (2 BCD-Stellen = 1 Byte) der Fakultät wird als Pointer auf die erstellte Multiplikationstabelle benutzt. Das niederwertige Byte (LL) wird direkt in die Fakultät eingetragen, das mittlere (MM) und höchstwertige Byte (HH) werden als Überlauf „gemerkt“ und bei der Berechnung der nächsten Doppelziffer berücksichtigt. Das ist im Prinzip die Methode der schriftlichen Multiplikation, nur daß die Berechnungen mit jeweils zwei Stellen durchgeführt werden. Wer die Multiplikationstabelle ( $0..99 \cdot 0..99$ ) auswendig kennt, kann diese Methode auch beim schriftlichen Rechnen benutzen. Er muß dazu nur zwei Überträge berücksichtigen.

Wenn der Kopf (das höchstwertige Byte) der Fakultät erreicht ist, werden die Überträge vorangestellt, und der Pointer wird entsprechend erhöht.

Dieses Verfahren wird so lange wiederholt, bis  $N$  erreicht ist. Danach erfolgt der Rücksprung zu Applesoft.

## Benutzung des Programms

Der Befehl „&P“ bewirkt den Ausdruck der berechneten Fakultät auf dem Bildschirm über die Monitor-Routinen PRBYTE und PRHEX. Die Ausgabe kann durch Drücken

einer beliebigen Taste unterbrochen und nach erneutem Tastendruck fortgesetzt werden.

Nach Eingabe von „&N“ ( $N = \text{Next}$ ) wird die nächsthöhere Fakultät berechnet. Will man die numerische Eingabe über eine Variable vornehmen, so sollte der Variablenname nicht mit „P“ oder „N“ beginnen, da dann direkt in die PRINT- oder NEXT-Routine gesprungen wird. Der numerische Ausdruck nach „&“ muß positiv sein und im Bereich  $0..9999$  liegen. Der Bereich ab  $\$2000$  für die berechnete Fakultät erlaubt es, den Vorgang der Berechnung über HGR mitzuverfolgen. Deutlich ist zu sehen, wie die Fakultät sehr schnell größer wird, während am Ende ein immer länger werdender Schwanz von Endnullen mitgeschleppt wird.

Ein kleines Applesoft-Programm berechnet die Anzahl der Endnullen einer Fakultät:

```
10 INPUT "N! ";N: X = N
20 I = INT (X / 5)
30 SU = SU + I
40 IF I >= 5 THEN X = I : GOTO 20
50 PRINT "Endnullen von ";N;"! = ";SU
```

Die Anzahl der Stellen einer Fakultät kann man durch Addition der dekadischen Logarithmen von 1 bis  $N$  ermitteln:

```
10 INPUT "N! ";N
20 KO = 1 / LOG (10)
30 FOR I = 1 TO N
40 SL = SL + LOG (I) * KO
50 NEXT
60 PRINT "Stellenzahl: "; INT(SL+1)
```

Soll das Assembler-Programm von einem Applesoft-Programm aus benutzt werden, sollten HIMEM auf  $9 * 4096$  und LOMEM auf  $6 * 4096 + 6 * 256$  gesetzt werden, da FAKULTAET den Bereich  $\$9000$  bis  $\$91FF$  belegt und die Multiplikationstabellen bei  $\$9200$  bis  $\$94FF$  ablegt. Bei einem nicht allzulangen Applesoft-Programm (Programmende höchstens  $\$1FFF$ ) sollte genügend Platz für die Variablen vorhanden sein. Der Ampersand-Vektor muß vom Monitor aus (3F6: 00 90) oder von Applesoft durch POKE 1014,0: POKE 1015,144 initialisiert werden.

Das Applesoft-Programm **FAKULTAET.-DEMO** zeigt ein Beispiel zur Benutzung der Fakultät-Routine.

## FAKULTAET

```

1 *****
2 *
3 * Fakultäten 0..9999
4 *
5 * Roland Fietkau
6 * Manfred Fietkau
7 * April 1985
8 *
9 *****
10 *
11 *          ORG $9000
12 *
13 * FAC -> String
14 FOUT EQU $ED34
15 * numerischen Ausdruck auswerten
16 FRMNUM EQU $DD67
17 * linkes Nibble drucken
18 PRHEX EQU $FDE3
19 * Byte drucken
20 PRBYTE EQU $FDDA
21 HOME EQU $FC58
22 * Character Output
23 COUT EQU $FDED
24 CHRGET EQU $00B1
25 KEY EQU $C000
26 STROBE EQU $C010
27 * Low-Byte der berechneten Fakultät
28 FAK EQU $2000
29 *
30 * ABCD! Low/high (dezimal)
31 NLO EQU $00
32 NHI EQU $01
33 * Pointer zum hochzählen
34 CLO EQU $02
35 CHI EQU $03
36 * Pointer auf Fakultätsende
37 PEND EQU $04
38 * Überlauf low/high
39 ULO EQU $06
40 UHI EQU $07
41 PCOUNT EQU $FE
42 *
43 * Multiplikationstabellen
44 MULTLO EQU $9200
45 MULTMI EQU $9300
46 MULTHI EQU $9400
47 *
48 *****
49 *
9000: C9 4E 50 *          CMP #'N'
51 *
52 * Nächste Fakultät berechnen
9002: F0 4B 53 *          BEQ NEXTFAK
9004: C9 50 54 *          CMP #'P'
55 * Fakultät ausgeben
9006: F0 41 56 *          BEQ DOPRINT
57 *
58 * Numerischen Ausdruck bei TXTPTR
59 * in FAC übertragen und durch
60 * FOUT in String bei $100.. umwandeln
61 * String dezimal in $00/$01
62 *
9008: 20 67 DD 63 *          JSR FRMNUM
900B: 20 34 ED 64 *          JSR FOUT
900E: A2 00 65 *          LDX #0
9010: BD 00 01 66 A1 *          LDA $100,X
9013: F0 04 67 *          BEQ A2
9015: E8 68 *          INX
9016: 4C 10 90 69 *          JMP A1
9019: 20 3C 90 70 A2 *          JSR FLOAT
901C: 85 00 71 *          STA NLO
901E: 20 3C 90 72 *          JSR FLOAT
9021: 0A 73 *          ASL
9022: 0A 74 *          ASL
9023: 0A 75 *          ASL
9024: 0A 76 *          ASL
9026: 05 00 77 *          ORA NLO
9027: 85 00 78 *          STA NLO
9029: 20 3C 90 79 *          JSR FLOAT
902C: 85 01 80 *          STA NHI
902E: 20 3C 90 81 *          JSR FLOAT
9031: 0A 82 *          ASL
9032: 0A 83 *          ASL
9033: 0A 84 *          ASL
9034: 0A 85 *          ASL
9035: 05 01 86 *          ORA NHI

```

```

9037: 85 01 87 *          STA NHI
9039: 4C BC 90 88 *          JMP COMP
89 *
903C: CA 90 *          FLOAT DEX
903D: 30 07 91 *          BMI F1
903F: BD 00 01 92 *          LDA $100,X
9042: 38 93 *          SEC
9043: E9 30 94 *          SBC #$30
9045: 60 95 *          RTS
9046: A9 00 96 *          LDA #0
9048: 60 97 *          RTS
98 *
9049: 20 6F 91 99 *          DOPRINT JSR PRINT
904C: 4C 6F 90 100 *          JMP ALLDONE
904F: AD BA 90 101 *          NEXTFAK LDA HOLDPLO
9052: 85 04 102 *          STA PEND
9054: AD BB 90 103 *          LDA HOLDPHI
9057: 85 05 104 *          STA PEND+1
9059: F8 105 *          SED
905A: A5 00 106 *          LDA NLO
905C: 85 02 107 *          STA CLO
905E: 18 108 *          CLC
905F: 69 01 109 *          ADC #1
9061: 85 00 110 *          STA NLO
9063: A5 01 111 *          LDA NHI
9065: 85 03 112 *          STA CHI
9067: 69 00 113 *          ADC #0
9069: 85 01 114 *          STA NHI
906B: D8 115 *          CLD
906C: 20 D7 90 116 *          JSR ST0
906F: 20 B1 00 117 *          ALLDONE JSR CHRGET
9072: 60 118 *          RTS
119 *
120 * Aktuelle Doppelziffer
121 * von FAK wird von 1 bis 99
122 * aufaddiert und in
123 * MULTLO/MI/HI übertragen
124 *
9073: A2 01 125 *          MULTTAB LDX #1
9075: 8E B6 90 126 *          STX COUNT
9078: A9 00 127 *          LDA #0
907A: 8D B7 90 128 *          STA HOLDLO
907D: 8D B8 90 129 *          STA HOLDMI
9080: 8D B9 90 130 *          STA HOLDHI
9083: F8 131 *          SED
9084: 18 132 *          CLC
9085: AD B7 90 133 *          LDA HOLDLO
9088: 65 02 134 *          ADC CLO
908A: 8D B7 90 135 *          STA HOLDLO
908D: 9D 00 92 136 *          STA MULTLO,X
9090: AD B8 90 137 *          LDA HOLDMI
9093: 65 03 138 *          ADC CHI
9095: 8D B8 90 139 *          STA HOLDMI
9098: 9D 00 93 140 *          STA MULTMI,X
909B: AD B9 90 141 *          LDA HOLDHI
909E: 69 00 142 *          ADC #0
90A0: 8D B9 90 143 *          STA HOLDHI
90A3: 9D 00 94 144 *          STA MULTHI,X
90A6: AD B6 90 145 *          LDA COUNT
90A9: 69 01 146 *          ADC #1
90AB: B0 07 147 *          BCS TABDONE
90AD: 8D B6 90 148 *          STA COUNT
90B0: AA 149 *          TAX
90B1: 4C 84 90 150 *          JMP MUL1
90B4: D8 151 *          TABDONE CLD
90B5: 60 152 *          RTS
153 *
90B6: 00 154 *          COUNT HEX 00
90B7: 00 155 *          HOLDLO HEX 00
90B8: 00 156 *          HOLDMI HEX 00
90B9: 00 157 *          HOLDHI HEX 00
90BA: 00 158 *          HOLDPLO HEX 00
90BB: 00 159 *          HOLDPHI HEX 00
160 *
161 * Initialisierung
162 *
90BC: A9 00 163 *          COMP LDA #0
90BE: 8D 00 92 164 *          STA MULTLO
90C1: 8D 00 94 165 *          STA MULTHI
90C4: 8D 00 93 166 *          STA MULTMI
90C7: 85 03 167 *          STA CHI
90C9: 85 04 168 *          STA PEND
90CB: A9 20 169 *          LDA #>FAK
90CD: 85 05 170 *          STA PEND+1
90CF: A0 00 171 *          LDY #0
90D1: A9 01 172 *          LDA #1
90D3: 91 04 173 *          STA (PEND),Y
90D5: 85 02 174 *          STA CLO

```

```

90D7: 20 ED 90 175 ST0 JSR TEST
90DA: 90 0B 176 BCC ST1
90DC: A5 04 177 LDA PEND
90DE: 8D BA 90 178 STA HOLDPLO
90E1: A5 05 179 LDA PEND+1
90E3: 8D BB 90 180 STA HOLDPHI
90E6: 60 181 RTS
90E7: 20 0B 91 182 ST1 JSR DOMULT
90EA: 4C D7 90 183 JMP ST0
184 *
185 * Testen, ob Pointer CLO/CHI
186 * auf NLO/NHI hochgezählt ist
187 *
90ED: A5 03 188 TEST LDA CHI
90EF: C5 01 189 CMP NHI
90F1: 90 08 190 BCC NOTYET
90F3: A5 02 191 LDA CLO
90F5: C5 00 192 CMP NLO
90F7: 90 02 193 BCC NOTYET
90F9: 38 194 SEC
90FA: 60 195 RTS
90FB: F8 196 NOTYET SED
90FC: A5 02 197 LDA CLO
90FE: 18 198 CLC
90FF: 69 01 199 ADC #1
9101: 85 02 200 STA CLO
9103: A5 03 201 LDA CHI
9105: 69 00 202 ADC #0
9107: 85 03 203 STA CHI
9109: D8 204 CLD
910A: 60 205 RTS
206 *
910B: A9 00 207 DOMULT LDA #0
910D: 85 FE 208 STA PCOUNT
910F: A9 20 209 LDA #>FAK
9111: 85 FF 210 STA PCOUNT+1
9113: 20 73 90 211 JSR MULTTAB
9116: A0 00 212 LDY #0
9118: 84 06 213 * STY ULO
911A: 84 07 214 STY UHI
911C: 18 215 DM1 CLC
911D: F8 216 SED
911E: B1 FE 217 LDA (PCOUNT), Y
9120: AA 218 TAX
9121: BD 00 92 219 LDA MULTLO, X
9124: 65 06 220 ADC ULO
9126: 91 FE 221 STA (PCOUNT), Y
9128: BD 00 93 222 LDA MULTMI, X
912B: 65 07 223 ADC UHI
912D: 85 06 224 STA ULO
912F: BD 00 94 225 LDA MULTHI, X
9132: 69 00 226 ADC #0
9134: 85 07 227 STA UHI
9136: D8 228 CLD
9137: A5 FE 229 LDA PCOUNT
9139: C5 04 230 CMP PEND
913B: D0 22 231 BNE WEITER
913D: A5 FF 232 LDA PCOUNT+1
913F: C5 05 233 CMP PEND+1
9141: D0 1C 234 BNE WEITER
9143: A5 07 235 LDA UHI
9145: D0 0D 236 BNE STEP2
9147: A5 06 237 LDA ULO
9149: D0 01 238 BNE STEP1
914B: 60 239 RTS
240 *
241 * Pointer auf Fakultätsende
242 * um 1 erhöhen
243 *
914C: 20 68 91 244 STEP1 JSR INCPEND
914F: A5 06 245 LDA ULO
9151: 91 04 246 STA (PEND), Y
9153: 60 247 RTS
248 *
249 * Pointer um 2 erhöhen
250 *
9154: 20 4C 91 251 STEP2 JSR STEP1
9157: 20 68 91 252 JSR INCPEND
915A: A5 07 253 LDA UHI
915C: 91 04 254 STA (PEND), Y
915E: 60 255 RTS
256 *
915F: E6 FE 257 WEITER INC PCOUNT
9161: D0 02 258 BNE WE1
9163: E6 FF 259 INC PCOUNT+1
9165: 4C 1C 91 260 WE1 JMP DM1
261 *

```

```

9168: E6 04 262 INCPEND INC PEND
916A: D0 02 263 BNE IN1
916C: E6 05 264 INC PEND+1
916E: 60 265 IN1 RTS
266 *
*****
268 *
269 * Ausgabe auf Bildschirm
270 * oder Printer
271 *
916F: 20 CB 91 272 PRINT JSR KOPF
9172: AD BA 90 273 LDA HOLDPLO
9175: 85 FE 274 STA PCOUNT
9177: AD BB 90 275 LDA HOLDPHI
917A: 85 FF 276 STA PCOUNT+1
917C: A0 00 277 LDY #0
917E: 20 58 FC 278 JSR HOME
9181: B1 FE 279 LDA (PCOUNT), Y
9183: C9 10 280 CMP #10
9185: B0 06 281 BCS PPI
9187: 20 E3 FD 282 JSR PRHEX
918A: 20 AC 91 283 JSR DECCOUNT
918D: B1 FE 284 PP1 LDA (PCOUNT), Y
918F: 20 DA FD 285 JSR PRBYTE
9192: 20 AC 91 286 JSR DECCOUNT
9195: 20 9B 91 287 JSR WAIT
9198: 4C 8D 91 288 JMP PPI
289 *
290 * Tastendruck unterbricht den
291 * Ausdruck, erneuter Tasten-
292 * druck setzt ihn fort
293 *
919B: 2C 00 C0 294 WAIT BIT KEY
919E: 10 0B 295 BPL WA2
91A0: 2C 10 C0 296 BIT STROBE
91A3: 2C 00 C0 297 WA1 BIT KEY
91A6: 10 FB 298 BPL WA1
91A8: 2C 10 C0 299 BIT STROBE
91AB: 60 300 WA2 RTS
301 *
91AC: A5 FE 302 DECCOUNT LDA PCOUNT
91AE: D0 0D 303 BNE DEC1
91B0: A5 FF 304 LDA PCOUNT+1
91B2: C9 20 305 CMP #>FAK
91B4: D0 07 306 BNE DEC1
91B6: 68 307 PLA
91B7: 68 308 PLA
91B8: A9 00 309 LDA #0
91BA: 85 22 310 STA $22
91BC: 60 311 RTS
91BD: A5 FE 312 DEC1 LDA PCOUNT
91BF: 38 313 SEC
91C0: E9 01 314 SBC #1
91C2: 85 FE 315 STA PCOUNT
91C4: A5 FF 316 LDA PCOUNT+1
91C6: E9 00 317 SBC #0
91C8: 85 FF 318 STA PCOUNT+1
91CA: 60 319 RTS
320 *
91CB: 20 58 FC 321 KOPF JSR HOME
91CE: A5 01 322 LDA NHI
91D0: 20 DA FD 323 JSR PRBYTE
91D3: A5 00 324 LDA NLO
91D5: 20 DA FD 325 JSR PRBYTE
91D8: A9 A0 326 LDA #A0
91DA: 20 ED FD 327 JSR COUT
91DD: A9 A1 328 LDA #"!"
91DF: 20 ED FD 329 JSR COUT
91E2: A9 A0 330 LDA #A0
91E4: 20 ED FD 331 JSR COUT
91E7: A9 BD 332 LDA #"-="
91E9: 20 ED FD 333 JSR COUT
91EC: A2 27 334 LDY #39
91EE: A9 8D 335 LDA #8D
91F0: 20 ED FD 336 JSR COUT
91F3: A9 AD 337 LDA #"-="
91F5: 20 ED FD 338 K1 JSR COUT
91F8: CA 339 DEX
91F9: 10 FA 340 BPL K1
91FB: A9 04 341 LDA #4
91FD: 85 22 342 STA $22
91FF: 60 343 RTS

```

512 Bytes



## FAKULTAET.DEMO

```

100 REM FAKULTAET.DEMO
110 REM -----
120 REM
130 PRINT CHR$(4)"BLOAD FAKULTAET"
140 POKE 1013,76: POKE 1014,0: POKE 1015,144:
    REM 3F5: 4C 00 90 = JMP $9000
150 A = 12 * 4096 + 5 * 16: REM $C050
160 HIMEM: 9 * 4096
170 HGR : TEXT
180 HOME : PRINT "G)rafik ":
190 IF G THEN INVERSE : PRINT "EIN": NORMAL : PRINT "/AUS"
    : GOTO 210
200 PRINT "EIN/": INVERSE : PRINT "AUS": NORMAL
210 PRINT "F)akultät berechnen"
220 PRINT "N)ächste Fak. berechnen"
230 PRINT "A)usgabe Bildschirm"
240 PRINT "E)nde"
250 PRINT : PRINT "-----"
260 PRINT "=> ": GET T$
270 IF T$ = "G" THEN G = NOT G: GOTO 180
280 IF T$ = "F" THEN INPUT "Fakultät von: ":FA: GOTO 330
290 IF T$ = "N" THEN 390
300 IF T$ = "A" THEN & P: GOTO 370
310 IF T$ = "E" THEN END
320 PRINT CHR$(7):: GOTO 180
330 IF G THEN POKE A,0: POKE A + 4,0: POKE A + 7,0
340 & FA
350 IF G THEN PRINT CHR$(7): CHR$(7):: GET T$: TEXT
360 GOTO 180
370 PRINT : PRINT "-----"
380 PRINT "<Taste>": GET T$: GOTO 180
390 IF G THEN POKE A,0: POKE A + 4,0: POKE A + 7,0
400 & N
410 GOTO 350
    
```

## Ein Beispielausdruck:

```

200! = 7886578673647905035523632139321850622951359776871732
63294742533244359449963403342920304284011984623904177212138
9196388302576427902426371050619266249528299311346285727076
3317237396988943922445621451664240254033291864131274282948
53277524242407573903240321257405579568660226031904170324062
3517008587961789222227896237038973747200000000000000000000
0000000000000000000000000000000000000000000000000000000000
    
```

```

1          ORG $0300
2          *
3          * Einfaches Dezimalmodus-
4          * Additionsbeispiel/US
5          *
6          * 199 + 1 = 200
7          *
8          SUMMAND1 HEX 0199
9          SUMMAND2 HEX 0001
10         SUMME    HEX 0000          ;0200
11         SED
12         CLC
13         LDA     SUMMAND1+1
14         ADC     SUMMAND2+1
15         STA     SUMME+1
16         LDA     SUMMAND1
17         ADC     SUMMAND2
18         STA     SUMME
19         CLD
20         RTS
28 Bytes
    
```



## MICROMINT

### VOLLTREFFER



**LASAR 16**  
IBM 256 K, 2 x TEAC B FDD, Contr. color Graphik, Multifunktionscard, Tastatur, Monitor  
Netzteil 15 A **4.678,-**

**LASAR ZE**  
- Apple comp. 64 K + 12 K ROM + 6502 + Z 80 A  
80 Z sw Tastatur **1.290,-**

**Außerdem volles Rückgaberecht innerhalb 14 Tagen ohne Begründung.**

	Apple	IBM
● Mehrzweckklappgehäuse lt. Abb.	147,-	147,-
● Schaltnetzteile Apple 5 A/IBM 15 A	115,-	238,-
● Profitastatur dtsh. LASAR 2000	291,-	291,-
● Interface ab	75,-	148,-
● Monitor 22 Mhz incl. Fuß, bernstein	289,-	289,-

Kaufgarantie/Tiefpreisgarantie/1A Qualität: 100 % kompatibel inkl. Systemsoftware  
Made by Micromint: Apple II 495,-, Apple IIe 695,-, IBM 895,- Fertigplatinen. Tragbare Gehäuse für 7-Zoll/9-Zoll-Monitore 595,- DM inkl. Tastaturen. Winchester 27 MB auf Anfrage 1A First Class Controller bis 140 MB 935,- DM.

**Generalimporteur MICROMINT Computer GmbH**  
Hochdahler Straße 151, 4006 Erkrath 2  
Telex 8589305 mcm

## ccp datentechnik

### 640 KByte-Drives für den Apple //c!!

- 5¼- od. 3½-Zoll-Format (Teac FD55/35-F)
- FD55-F umschaltbar auf 35/40 Track
- Anschluß an die externe Laufwerkbuchse
- Durch Einbauplatine (kein Löten) 640 KByte im Direktzugriff
- Einfache Anpassung für DOS 3.3, UCSD-Pascal und PRODOS durch menügeführten Patch
- Anpassung von CP/M in Verbindung mit einer Z 80-Zusatzplatine in Vorbereitung
- anschlussfertig im Gehäuse . . . . . **DM 1090,-**

### Festplatten für Apple II (//e)

- 5¼ Zoll-Format (Slimline)
- Booten direkt von der Festplatte in DOS 3.3, UCSD-Pascal, PRODOS und CP/M 2.2 / 3.0
- Gemischtbetr. mit 35/40/80/160 Track-Drives
- Copy- und Install-Programme im Lieferumfang
- Umfangreiches Manual
- z. B. 12 MB form. incl. Netzteil u. Contr., anschlussfertig an Ihren Apple . . . . . **DM 3835,-**

### 640 KByte-Drives für Apple II (//e)

- 5¼- od. 3½-Zoll-Format (Teac FD55/35-F)
- FD55-F umschaltbar auf 40 Track (Apple kompatibel)
- Installationssoftware für DOS 3.3, UCSD-Pascal, CP/M 2.2, CP/M 2.23 (60K), PRODOS, AP22, ALS CP/M+
- Umfangreiches Handbuch
- Anschlussfertige Auslieferung incl. Contr. und 2 Drives
- Diskstation 55II (2 Teac FD55-F, 1.2 MB) . . . . . **DM 1598,-**
- Diskstation 35II (2 Teac FD35-F, 1.2 MB) . . . . . **DM 1580,-**

### 80 Zeichen + 64 K für Apple //e

- und jetzt einsetzen . . . . . **DM 158,-**

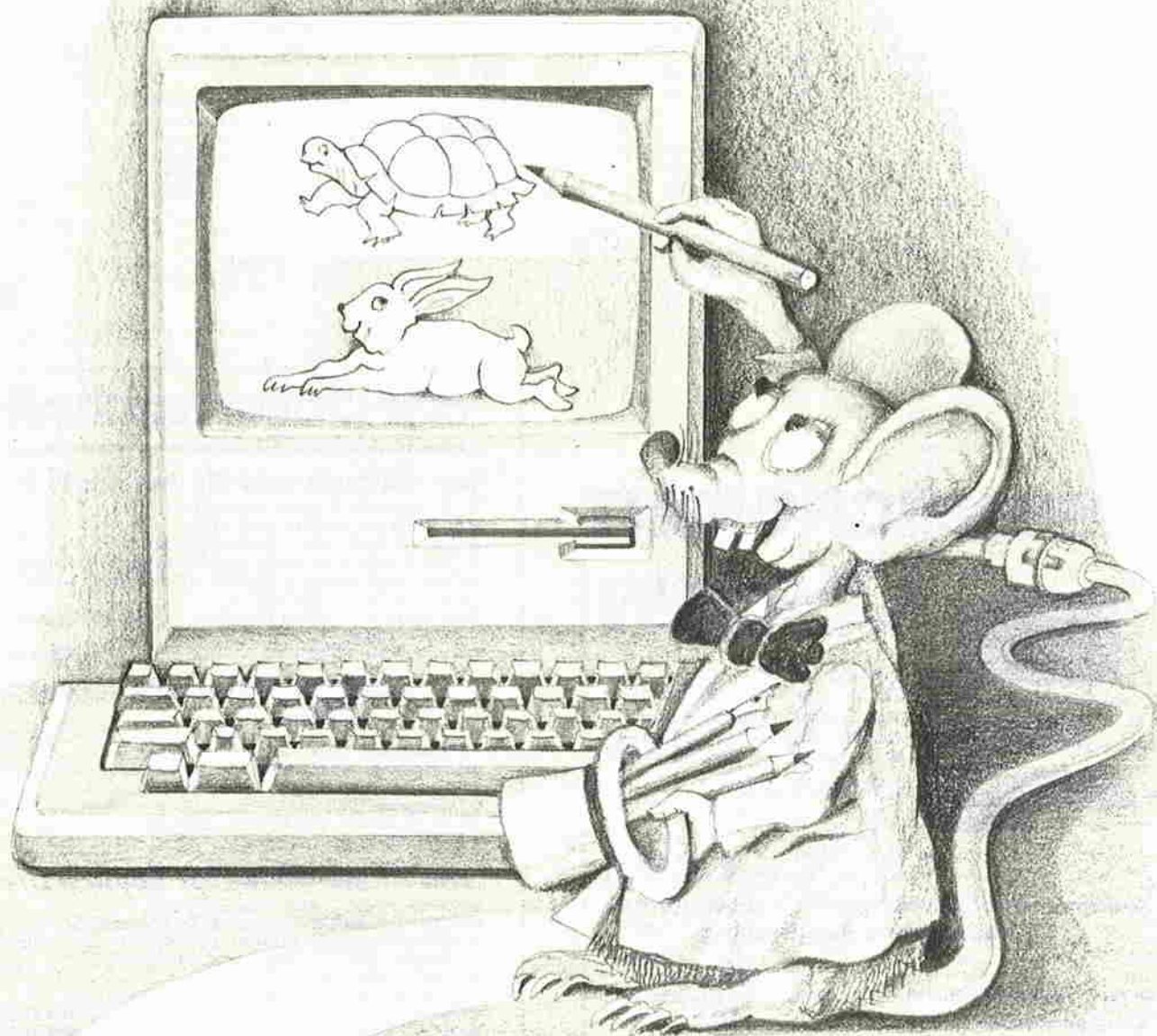
## Alles für Ihren Apple

Info bei:  
**ccp-datentechnik**  
Herderstraße 12 - 2000 Hamburg 76  
Telefon 040/225676

# Microsoft Basic leicht geMAChT

Teil 4: Die Grafik

von Pit Captain



## 7. GRAFIK BEIM MS-BASIC

Da beim Macintosh die Grafik eine sehr große Rolle spielt, erwartet man auch vom Microsoft Basic relativ komfortable und leistungsfähige Grafik-Befehle.

Dies ist leider nicht der Fall. Das Basic bietet – im Vergleich zu den Möglichkeiten des Mac – nur sehr wenige Grafik-Befehle. Dieser Mangel wird dadurch etwas ausgeglichen, daß man Zugriff auf einige im ROM eingebaute Routinen hat.

Mit diesen Routinen hat man dann etwas mehr Möglichkeiten, Grafik zu erstellen. Doch ist die Lösung mit den ROM-Routinen nicht sehr elegant, zumal deren Benutzung im Handbuch nur sehr kurz beschrieben wird.

### 7.1. Basic-Befehle

#### 7.1.1. Koordinaten

Bei allen Grafik-Befehlen müssen Koordinaten angegeben werden, die bestimmen, wo im *Ausgabefenster* gearbeitet wird. Diese Koordinaten kann man auf zwei Arten angeben:

**absolut:** (X,Y) bestimmt den Punkt im Fenster, der die angegebenen Koordinaten besitzt. Die erste Zahl ist die X-, die zweite Zahl die Y-Koordinate. Beispiel: (100,25) – die linke, obere Ecke hat die Koordinaten (0,0).

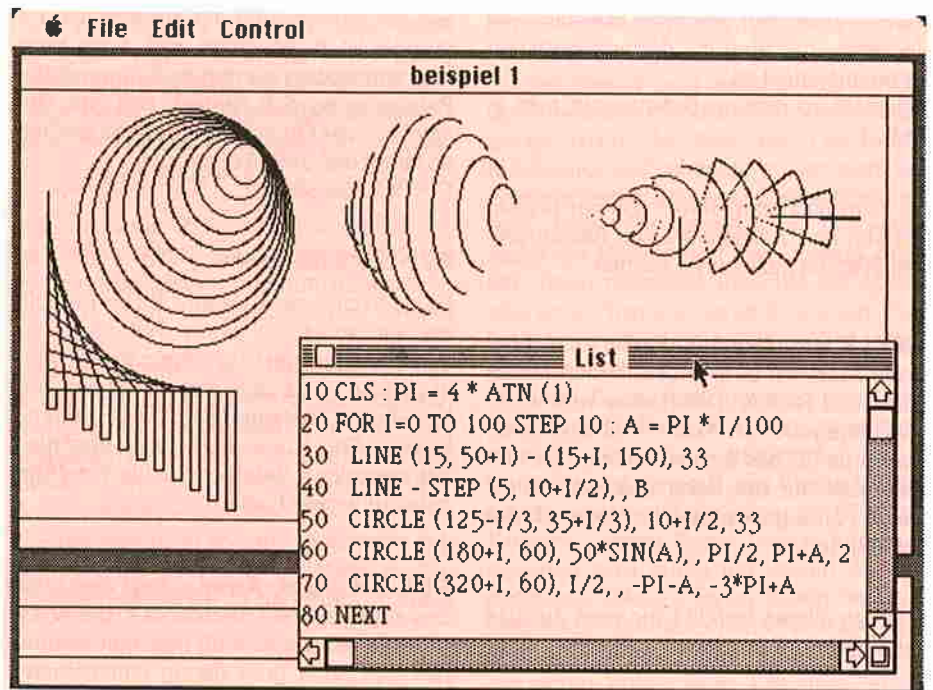


Abb. 1

**relativ:** (DX,DY) bestimmt den Punkt im Fenster, der vom zuletzt angesprochenen Punkt so weit entfernt ist, wie es die Koordinaten angeben. Beispiel: angenommen, es wurde zuletzt der Punkt (100,25) gezeichnet. Dann bezeichnet „STEP (-20, 30)“ den Punkt (80,55).

#### 7.1.2. Farben

Bei fast allen Grafik-Befehlen kann man eine Farbe angeben, mit der gezeichnet werden soll. Jede Farbe wird durch eine bestimmte Nummer dargestellt. Die beim Macintosh vorgesehenen Farben sind mit ihren zugehörigen Nummern in der **Tabelle** angegeben.

Der Mac kann zur Zeit nur schwarz und weiß darstellen. Alle anderen Farben werden als schwarz interpretiert (vgl. dazu die Spalte „Bit 0“ in der Tabelle).

Falls man bei den Befehlen keine Farbe angibt, setzt das Basic als Defaultwert schwarz ein.

#### 7.1.3. Einzelne Punkte

**PSET Koord., Farbe** und **PRESET Koord., Farbe** – zeichnen den angegebenen Punkt in einer bestimmten Farbe. Die beiden Befehle unterscheiden sich nur dann, wenn man *keine* Farbe angibt. PSET benutzt dann die sogenannte

„Vordergrund“-Farbe (normalerweise schwarz), dagegen benutzt PRESET die „Hintergrund“-Farbe (normalerweise weiß).

#### 7.1.4. Linien und Flächen

Mit dem Befehl „LINE“ (nicht zu verwechseln mit der gleichnamigen ROM-Routine (!), s.u.) kann man Linien und Rechtecke zeichnen:

**LINE Anf - End, Farbe** – zeichnet eine Linie von der Anfangs- zur End-Koordinate in der angegebenen Farbe.

Man kann die erste Koordinate weglassen; dann beginnt die Linie beim zuletzt angesprochenen Punkt.

Wenn keine Farbe angegeben wird, so wird in schwarz gezeichnet. Der Befehl: LINE (100,25) - STEP (-20,30) zeichnet also eine schwarze Linie vom Punkt (100,25) zum Punkt (80,55).

An den „LINE“-Befehl kann man noch zwei Zusätze anhängen:

**, B** – (Box) zeichnet ein Rechteck, bei dem zwei gegenüberliegende Ecken durch die angegebenen Koordinaten bestimmt werden. Es wird nur der Rahmen des Rechtecks in der angegebenen Farbe gezeichnet, das Innere bleibt unverändert.

Farben	Bits		Hex.	Dez.
	876 5 432 10			
Schwarz	000 1 000 01		\$021	33
Weiß	000 0 111 10		\$01E	30
Rot	011 0 011 01		\$0CD	205
Grün	101 0 101 01		\$155	341
Blau	110 0 110 01		\$199	409
Cyan	100 0 100 01		\$111	273
Magenta	010 0 010 01		\$089	137
Gelb	001 0 001 01		\$045	69

Normal  
 Invers  
 Blau } additiv für Monitor  
 Grün }  
 Rot }  
 Schwarz  
 Gelb } subtraktiv für Drucker  
 Magenta }  
 Cyan }

Tabelle der Farbwerte



, **BF** – (Box Fill) zeichnet ebenfalls ein Rechteck, das aber mit der angegebenen Farbe ausgefüllt ist.

Beispiele zu diesem Befehl sieht man in **Abb. 1**.

Mit dem nächsten Befehl können Kreise, Ellipsen und Ausschnitte aus diesen beiden Flächen gezeichnet werden:

**CIRCLE Mitte, Radius, Farbe** – zeichnet einen Kreis mit dem angegebenen Mittelpunkt und Radius. Der Radius wird in Pixeln angegeben (1 Pixel = 1 Grafikpunkt). Zur Farbe gilt das bisher Gesagte. Es wird wiederum nur der Rand des Kreises mit dieser Farbe gezeichnet, das Innere bleibt unverändert.

Auch an diesen Befehl kann man Zusätze anhängen:

, **Anf, End** – bestimmt einen Ausschnitt des Kreises, der gezeichnet werden soll. Der Kreis wird vom Winkel **Anf** bis zum Winkel **End** gezeichnet. Die Winkel werden im Bogenmaß angegeben, wobei der Winkel 0 nach rechts zeigt, der Winkel  $\pi/2$  nach unten.

Falls einer der beiden Winkel negativ ist, so wird die Randlinie des Kreisabschnitts mit dem Mittelpunkt des Kreises verbunden, so daß eine Art „Kuchenstück“ entsteht. Von den Winkeln wird der *Betrag* genommen. (Dies entspricht also *nicht* der Addition von  $2 * \pi$ !)

Leider wird keine *Linie* zum Mittelpunkt gezogen, wie man vielleicht annehmen könnte, sondern es wird ein *Kreisabschnitt* von 2 Grad gezeichnet. Dieser ist dann natürlich außen breiter als innen, wo er fast gar nicht zu sehen ist.

Beispiele zu diesem Befehl sieht man ebenfalls in **Abb. 1**.

Es gibt noch einen weiteren Parameter, den man anfügen kann:

, **Quotient** – gibt das Verhältnis vom Radius in Y-Richtung zum Radius in X-Richtung an. Damit ist es möglich, Ellipsen zu zeichnen.

Der im Befehl angegebene Radius ist immer der *größere* von den beiden Radien. Falls der Quotient kleiner als 1 ist, ist also der Radius in X-Richtung angegeben, ansonsten der in Y-Richtung.

Beispiele dazu sind ebenfalls in der **Abb. 1** zu sehen.

Bei der Verwendung der zusätzlichen Parameter ist zu beachten, daß durch Kommas angegeben werden muß, um welchen Parameter es sich handelt. Soll zum Beispiel nur der Quotient angegeben werden, so lautet der Befehl etwa:  
CIRCLE (50,50), 30, , , , 2

### 7.1.5 Sonstige Befehle

Es gibt noch zwei weitere Basic-Befehle, die es erlauben, einen Teil des Bildschirms (genauer: des Ausgabefensters) in eine Array-Variable einzulesen und später wieder auszugeben. Diese Befehle heißen „GET“ und „PUT“, sind aber nicht mit denselben Befehlen für die Ein-/Ausgabe zu verwechseln.

**GET Anf - End, Array** – liest den Inhalt des angegebenen Rechtecks in die Array-Variable ein. Dabei muß beachtet werden, daß das Array groß genug dimensioniert wurde, um auch alle Bits speichern zu können.

Jede Zeile des Bildes wird in Vielfachen von 16 Bits gespeichert. Falls das Bild etwa 21 Pixel breit ist, werden für jede Zeile 32 Bits oder 4 Bytes benötigt.

Damit läßt sich die Größe eines Bildes berechnen. Sei (X1,Y1) die linke, obere und (X2,Y2) die rechte, untere Ecke des Rechtecks, so muß das Array mindestens  $4 + (Y2 - Y1 + 1) * 2 * \text{INT}((X2 - X1 + 16)$

/ 16) Bytes lang sein. Der Ausdruck ab „2 \* INT“ gibt an, wieviele Bytes für eine Zeile benötigt werden. Dieser Wert wird mit der Zahl der Zeilen multipliziert. Die „4“ am Anfang des Ausdrucks kommt daher, daß neben dem Bild auch dessen Dimensionen (Breite und Höhe) gespeichert werden, die je 2 Bytes benötigen.

Für den Befehl:

GET (80,25) - (100,55), A%

muß A% mindestens 128 Bytes lang sein. Wieviele *Elemente* A% enthalten muß, sieht man, wenn man sich nochmals die Größen der einzelnen Zahlen-Typen (vgl. Teil 1, Pecker 2/84) vor Augen hält:

2 Bytes für ganze Zahlen,

4 Bytes für einfach genaue,

8 Bytes für doppelt genaue rationale Zahlen.

Das Array A% (ganze Zahlen) muß also mindestens 64 Elemente enthalten, was durch „DIM A% (63)“ erreicht wird.

**PUT Anf - End, Array, Modus** – gibt das im Array gespeicherte Bild wieder auf dem Bildschirm an der Koordinate **Anf** aus. („Anf“ ist die Koordinate der linken, oberen Ecke).

Wenn die Koordinate der rechten, unteren Ecke **End** nicht angegeben ist, so wird das Bild mit den ursprünglichen Dimensionen gezeichnet (s.o.).

Falls aber die zweite Koordinate angegeben ist, wird das Bild so gedehnt oder

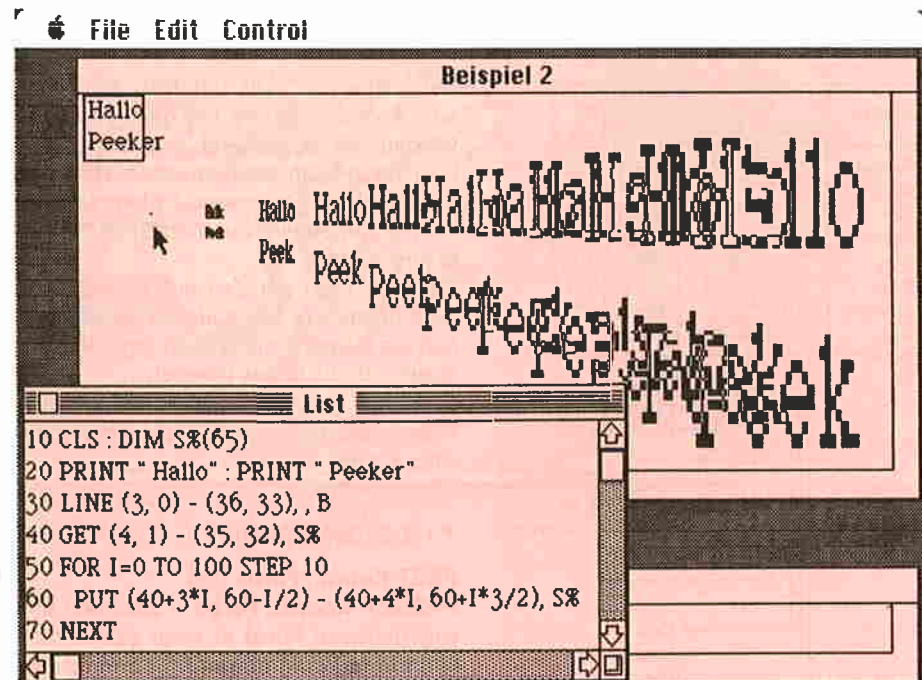


Abb. 2

gestaucht, daß es in das angegebene Rechteck hineinpaßt!

Der Modus, den man hinter dem Befehl angeben kann, bestimmt die Art, wie das Bild gezeichnet werden soll. Es kann eines der folgenden Worte angegeben werden:

**PSET** kopiert das Bild normal auf den Bildschirm.

**PRESET** kopiert das *Inverse* des Bildes auf den Bildschirm.

**AND, OR** oder **XOR** verknüpfen das Bild bitweise mit dem Bildschirminhalt.

Falls kein Modus angegeben ist, wird „XOR“ angenommen. Mit diesem Modus kann ein gezeichnetes Bild wieder gelöscht werden, wobei der Hintergrund unverändert bleibt.

Beispiele für „GET“ und „PUT“ sieht man in **Abb. 2**.

## 7.2. ROM-Routinen

### 7.2.1. Aufruf

Die im ROM des Macintosh vorhandenen Zeichenroutinen werden wie alle Programme in Maschinensprache mit dem Befehl „CALL“ aufgerufen (vgl. Teil 2, Pecker 1/2-85). An die meisten Routinen müssen Parameter wie z.B. die Koordinaten eines Rechtecks übergeben werden. Der Aufruf mit Parametern sieht allgemein so aus:

#### CALL Name (Par1, Par2, ...)

Die Namen der Routinen sind reservierte Wörter, die man wie die normalen Basic-Befehle nicht als Variablennamen benutzen darf.

### 7.2.2. Parameter

Da das ROM des Macintosh nichts über Basic-Variablen und deren Aufbau weiß, müssen die Parameter der einzelnen Aufrufe vom Basic aus simuliert werden, was in Basic weitaus weniger elegant erscheint als in Maschinensprache oder in Pascal.

Die von MS-Basic aus erreichbaren Zeichenroutinen benötigen verschiedene Arten von Parametern:

**Word** (2 Bytes) gibt ganze Zahlen an (16 Bits, vorzeichenbehaftet). Damit werden z.B. einzelne Koordinaten dargestellt. Beim Aufruf braucht man hier nur eine ganze Zahl einzusetzen.

**Point** (2 Words = 4 Bytes) gibt die Koordinaten eines Punktes an.

Beim Aufruf ist es am einfachsten, wenn man nicht *einen* Parameter mit 4 Bytes angibt, sondern *zwei* Parameter mit je 2 Bytes. In diesem Fall kann man die Koordinaten mittels zweier ganzer Zahlen (z.B. „(X%, Y%)“) angeben.

Alle sonstigen Parameterarten benötigen mehr als 4 Bytes. Sie werden nicht direkt an das Maschinenprogramm übergeben, sondern nur ihre Adresse. Diese besteht aus 4 Bytes.

Falls ein Parameter von der ROM-Routine an das Basic-Programm zurückgegeben wird (z.B. die Position des „Pen“, s.u.), so wird ebenfalls nur die Adresse des Parameters übergeben, egal, ob der Parameter länger als 4 Bytes ist oder nicht.

Wer das jetzt nicht so ganz verstanden hat, sieht sich am besten die Beispiele zu den einzelnen Routinen in den **Abb. 2 bis 4** an.

Es gibt folgende Parameter, die länger als 4 Bytes sind:

**Rect** (4 Words = 8 Bytes) gibt die Koordinaten eines Rechtecks an in der Reihenfolge oberer, linker, unterer und rechter Rand.

Beim Aufruf wird folgendes übergeben: „VARPTR (R%(0))“, wobei R% ein Array mit mindestens 4 Elementen (ganzen Zahlen) ist. Die Funktion „VARPTR“ liefert gerade die Adresse der Array-Elemente.

**Pattern** (8 Bytes) gibt ein Muster an, mit dem Flächen gefüllt werden können (wie z.B. bei „MacPaint“ die Muster in der unteren Zeile).

Ein solches Muster besteht aus 8 \* 8 Pixeln, benötigt also 8 Bytes. Das erste Byte entspricht dabei der obersten Zeile des Musters, das zweite Byte der zweiten Zeile, usw. In jeder Zeile entspricht das Pixel ganz links dem höchstwertigen Bit im Byte oder dem Zahlenwert 128. Das Pixel ganz rechts entspricht demnach dem niederwertigsten Bit im Byte oder dem Zahlenwert 1.

Der Parameter beim Aufruf lautet dann etwa „VARPTR (P%(0))“, wobei P% wieder ein Array mit mindestens 4 Elementen sein muß.

**Cursor** (68 Bytes) bestimmt die Gestalt des Cursors, also das Bild, das mit der Maus bewegt wird (z.B. der Pfeil).

Die ersten 32 Bytes stellen die sogenannten „Cursordaten“ dar. Dies ist ein Rechteck aus 16 \* 16 Pixeln, das das Bild des Cursors angibt.

Doch werden normalerweise nicht alle diese Pixel gezeichnet, da sonst der Cursor

immer rechteckig wäre. Welche Pixel gezeichnet werden sollen, bestimmt die „Cursormaske“, die aus den nächsten 32 Bytes besteht. Es handelt sich hier wieder um ein 16\*16-Rechteck. Wenn ein Bit der Maske auf „1“ gesetzt ist, so wird das entsprechende Bit der Cursordaten auf dem Bildschirm gezeichnet, ansonsten bleibt der Hintergrund des Cursors sichtbar. Beim normalen Pfeil hat die Maske ebenfalls Pfeilform, ist aber um ein Pixel breiter als die Cursordaten. Dies ergibt einen weißen Rand um den Pfeil.

Mit dem Cursor will man nun nicht eine Fläche von 16 \* 16 Pixeln auswählen, sondern einen ganz bestimmten Punkt. Beim Pfeil ist dies z.B. die Pfeilspitze, bei einem kreuzförmigen Cursor die Mitte des Kreuzes. Dieser Punkt (auch „HotSpot“ genannt) wird durch die letzten 4 Bytes bestimmt. Diese 4 Bytes haben den Aufbau von „Point“ (s.o.). Ein Wert von (0,0) steht für die linke, obere Ecke, (15,15) für die rechte, untere Ecke. Die erste Zahl ist die Y-, die zweite die X-Koordinate.

Beim Aufruf wird folgendes übergeben: „VARPTR (C%(0))“, wobei C% ein Array mit mindestens 34 Elementen sein muß.

Nach diesen sehr theoretischen Ausführungen kommen wir nun zu den einzelnen Routinen und damit auch zu Beispielen, an denen das vorher Gesagte mit etwas mehr Leben erfüllt werden soll.

### 7.2.3. Cursor-Routinen

CALL **HideCursor** – diese Routine hat keine Parameter. Bei jedem Aufruf der Routine wird eine Variable des Betriebssystems, der sogenannte „CursorLevel“, um 1 erniedrigt.

Der „CursorLevel“ ist eine vorzeichenbehaftete 16-Bit-Zahl. Wenn diese Zahl negativ ist, ist der Cursor nicht sichtbar.

Mit einem Aufruf von „HideCursor“ erreicht man also, daß der Cursor nicht mehr zu sehen ist. (Dies kommt z.B. beim Booten des Mac vor, wenn das Fenster mit der Meldung „Willkommen zu Macintosh“ erscheint.)

CALL **ShowCursor** – auch diese Routine hat keine Parameter. Sie ist das Gegenstück zur „HideCursor“-Routine, denn sie erhöht die Variable „CursorLevel“ um 1.

Falls dadurch die Variable positiv (also größer oder gleich 0) wird, dann wird der Cursor wieder sichtbar. Zu beachten ist dabei, daß der „CursorLevel“ vom Betriebssystem her nie größer als Null wird. So hat z.B. die Folge:



CALL ShowCursor  
 CALL ShowCursor  
 CALL ShowCursor  
 CALL HideCursor

die Wirkung, daß der CursorLevel auf -1 steht und der Cursor also nicht sichtbar ist. CALL **ObscureCursor** – ebenfalls eine Routine ohne Parameter. Sie macht folgendes:

Zunächst ruft sie „HideCursor“ auf, das heißt, daß der Cursor verschwindet. Anschließend wird in einer bestimmten Variablen des Betriebssystems notiert, daß der Cursor mit „ObscureCursor“ versteckt wurde.

Sobald der Benutzer danach die Maus und damit den Cursor bewegt, wird sofort die Routine „ShowCursor“ aufgerufen; der Cursor wird wieder sichtbar.

Die Aufgabe von „ObscureCursor“ ist also, den Cursor solange verschwinden zu lassen, bis die Maus bewegt wird. Auch diese Routine wird manchmal beim Mac verwendet, z.B. verschwindet beim Microsoft Basic der Pfeil, sobald eine Taste gedrückt wird, wenn gerade kein Programm läuft. Der Pfeil wird allerdings sofort wieder sichtbar, wenn man die Maus bewegt.

CALL **InitCursor** – diese Routine hat keine Parameter. Sie bewirkt, daß der Cursor die gewohnte Form eines Pfeils bekommt, der nach links oben zeigt. Außerdem wird der „CursorLevel“ (s.o.) auf 0 gesetzt, so daß der Cursor sichtbar wird.

CALL **SetCursor** (VARPTR (C%(0))) – diese Routine verändert die Form des Cursors so, wie im Parameter C% angegeben. Der Aufbau und die Länge von C% wurde weiter oben schon erklärt.

Ein Beispiel zu dieser Routine ist in **Abb. 3** gegeben. Das Programm liest die neue Gestalt des Cursors aus „DATA“-Zeilen ein und verändert den Cursor entsprechend. Der neue Cursor ist im Bild in der Ecke des großen Quadrats zu sehen.

#### 7.2.4. Textausgabe

Es gibt 4 Routinen, mit denen die Art, wie Text ausgegeben wird, verändert werden kann. Da in jedem geöffneten Fenster eine andere Art der Textausgabe möglich ist, wird mit diesen Routinen nur die Ausgabeart des *aktiven* Fensters verändert. Bei einem laufenden Basic-Programm ist dies das Ausgabefenster. Die anderen Fenster (z.B. das Listfenster) werden dadurch nicht verändert.

CALL **TextSize** (SIZE) – verändert die Größe der Buchstaben. Jeder Text, der nach diesem Aufruf ausgegeben wird, hat

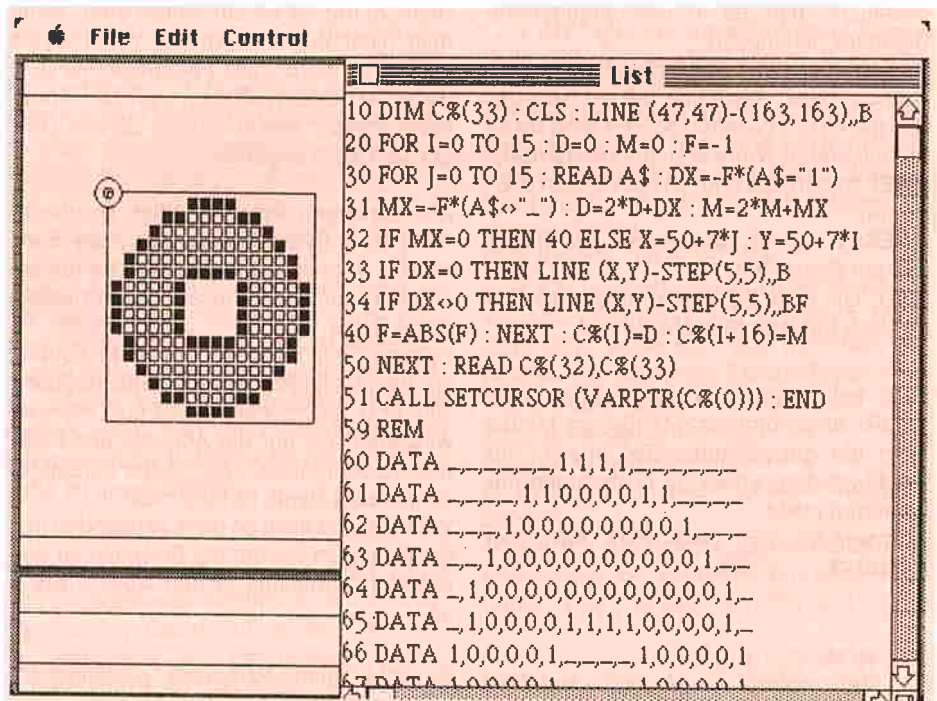


Abb. 3

die Größe SIZE. Diese Routine wird z.B. dann aufgerufen, wenn man etwa in „MacWrite“ eine andere Zeichengröße wählt.

CALL **TextFont** (FONT) – wählt einen neuen Zeichensatz aus. FONT ist eine Nummer, die den Zeichensatz bestimmt. Es gibt die folgenden Nummern:

- 0 – System-Zeichensatz
- 1 – normaler Basic-Zeichensatz
- 2 – New York
- 3 – Geneva
- 4 – Monaco (nicht proportional!)
- 5 – Venice
- 6 – London
- 7 – Athens
- 8 – San Francisco
- 9 – Toronto

CALL **TextFace** (FACE) – bestimmt die Art, wie die einzelnen Buchstaben ausgegeben werden. FACE ist dabei eine Zahl, deren einzelne Bits die folgende Bedeutung haben:

- Bit 0 (01) – Bold (fett)
- Bit 1 (02) – Italic (kursiv)
- Bit 2 (04) – Underline (unterstrichen)
- Bit 3 (08) – Outline (umrahmt)
- Bit 4 (16) – Shadow (mit Schatten)
- Bit 5 (32) – Condense (Schmalschrift)
- Bit 6 (64) – Extend (breit)

Die einzelnen Bits können wahlweise kombiniert werden (wobei nicht alle Kombinationen gut zu entziffern sind). Wenn z.B. FACE = 70 ist, so wird nach dem Aufruf jeder Text in breiter Schrift, unterstrichen und zudem kursiv ausgegeben. Auch diese Wahlmöglichkeiten kennt man z.B. von „MacWrite“.

CALL **TextMode** (MODE) – bestimmt die Art, wie der auszugebende Text mit dem Bildschirminhalt kombiniert wird. Es gibt die folgenden Werte:

- 0 – Copy (direkte Kopie)
- 1 – Or (normale Ausgabeart)
- 2 – Xor (s.o.)
- 3 – Bic (s.u.)

Die Ausgabeart „Bic“ bedeutet, daß die Buchstaben in weiß gezeichnet werden. Welche Wirkung diese einzelnen Arten bei unterschiedlichem Hintergrund haben, zeigt die **Abb. 4**.

#### 7.2.5 Der Pen

Der sogenannte „Pen“ ist der eigentliche Zeichenstift des Macintosh. Jedes Fenster besitzt seinen eigenen Stift. Dieser Stift hat eine ganze Reihe von Eigenschaften:

**PnLoc** gibt die Position des Stifts im jeweiligen Fenster an. Dies ist gleichzeitig die Position, an der der nächste Text ausgegeben wird.



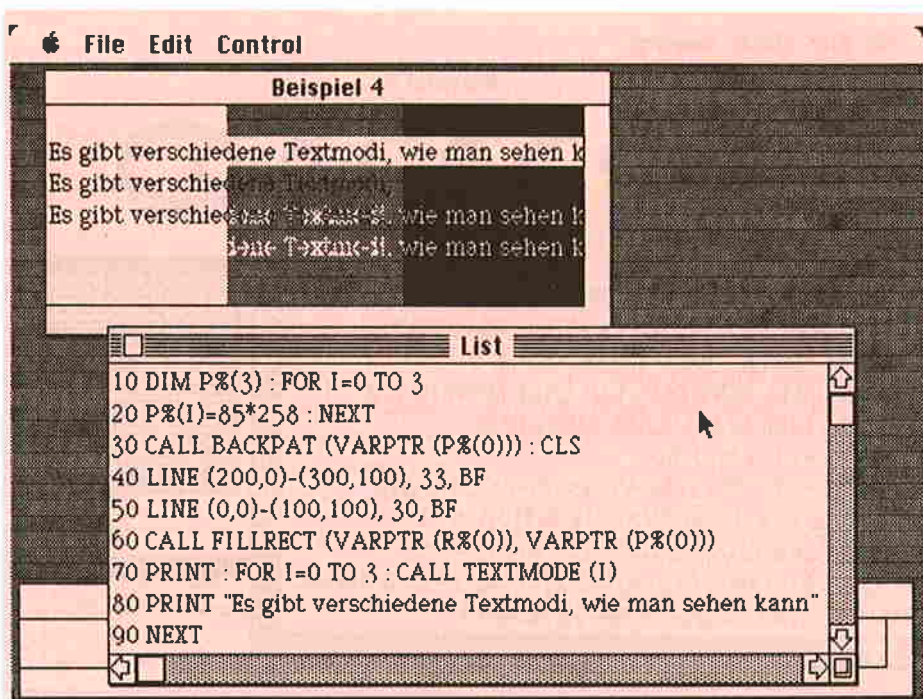


Abb. 4

**PnSize** gibt die Größe des Stifts an. Der Stift hat eine rechteckige Form, deren Höhe und Breite verändert werden kann. „PnLoc“ gibt dabei die Position der linken, oberen Ecke des Stifts an.

**PnMode** gibt die Art an, wie die „Farbe“ des Stifts mit dem Bildschirm kombiniert werden soll (vgl. oben die Routine „Text-Mode“).

**PnPat** stellt die „Farbe“ des Stifts dar. Es handelt sich um ein „Pattern“ (Muster aus 8 \* 8 Pixeln).

**PnVis** gibt an, ob der Stift sichtbar ist oder nicht. Falls hier ein negativer Wert vorhanden ist, dann ist der Stift nicht sichtbar (vgl. „CursorLevel“ weiter oben). Dies hat die Wirkung, daß alles, was mit dem Stift gezeichnet wird, nicht sichtbar ist.

Alle diese Eigenschaften können mit den nachfolgenden Routinen verändert werden:

CALL **HidePen** – erniedrigt „PnVis“ um 1. Dadurch wird der Pen unsichtbar.

CALL **ShowPen** – erhöht „PnVis“ wieder um 1. Dies ist also das Gegenstück zu „HidePen“.

Im Gegensatz zu „CursorLevel“ kann „PnVis“ auch größer als Null werden, so daß man hier aufpassen muß, daß man von beiden Routinen dieselbe Anzahl verwendet.

CALL **GetPen** (VARPTR (L%(0))) – gibt die momentane Position des Zeichenstifts in die Variable L% zurück. L% ist dabei ein Array mit mindestens zwei Elementen, wobei L%(0) die Y-Koordinate und L%(1) die X-Koordinate ist.

Dies ist die einzige ROM-Routine, die eine Information an das Basic zurückgibt. Deshalb muß auch hier die Funktion „VARPTR“ verwendet werden, obwohl der Parameter selbst vom Typ „Point“, also nur 4 Bytes lang ist.

CALL **PenSize** (B, H) – bewirkt, daß der Stift ein Rechteck mit der Breite B und Höhe H wird. Beide Parameter haben die Einheit Pixel.

CALL **PenMode** (MODE) – setzt „Pn-Mode“ auf den Wert von MODE. Die möglichen Werte sieht man in der obigen Tabelle, die bei „TextMode“ angegeben ist. **ACHTUNG:** Bei „PenMode“ muß zu jedem dieser Werte noch 8 addiert werden! MODE liegt demnach im Bereich 8 bis 11, wobei z.B. 10 die Bedeutung „Xor“ hat (10 = 8 + 2).

(Für die interessierten Leser: dies hängt damit zusammen, daß mit dem Pen kein Text ausgegeben wird, sondern ein Muster, nämlich „PnPat“. Bei der Ausgabe von Text muß MODE eine Zahl kleiner als 8 sein, bei der Ausgabe eines Musters dagegen größer als 7.)

CALL **PenPat** (VARPTR (P%(0))) – verändert „PnPat“, also das Muster des Stifts, zu dem Muster, das in P% angegeben ist. P% ist vom Typ „Pattern“, der weiter oben erklärt wurde.

CALL **PenNormal** – bringt den Stift wieder in seinen „normalen“ Zustand. Dabei werden die folgenden Aktionen ausgeführt: „PnSize“ wird auf (1,1) gesetzt (1 Pixel breit, 1 Pixel hoch).

„PnMode“ wird auf „Copy“, also auf 8 gesetzt.

„PnPat“ wird auf schwarz gesetzt, d.h. daß alle 8 \* 8 Bits auf 1 sind.

Die beiden anderen Eigenschaften des Stifts, „PnLoc“ und „PnVis“, bleiben unverändert.

CALL **MoveTo** (X, Y) – bewegt den Stift ohne zu zeichnen zu dem Punkt mit den angegebenen (absoluten) Koordinaten.

CALL **Move** (DeltaX, DeltaY) – bewegt den Stift ohne zu zeichnen horizontal um DeltaX und vertikal um DeltaY weiter (relative Koordinaten).

CALL **LineTo** (X, Y) – wie „MoveTo“, nur wird diesmal bei der Bewegung eine Linie gezeichnet. Dabei werden nun die Eigenschaften des Stifts (Muster, Größe, usw.) sichtbar.

CALL **Line** (DeltaX, DeltaY) – wie „LineTo“, nur sind die Koordinaten relativ angegeben.

Diese Routine hat leider denselben Namen wie der Basic-Befehl „LINE“ (s.o.).

Beispiele zu diesen Routinen sieht man in der **Abb. 5**.

### 7.2.6. Der Hintergrund

Es gibt eine Routine, mit der man das Muster verändern kann, mit dem normalerweise der Hintergrund ausgefüllt wird.

CALL **BackPat** (VARPTR (P%(0))) – bewirkt, daß fortan der Hintergrund mit dem Muster in P% (Pattern) ausgefüllt wird.

Dieses Muster wird z.B. dann verwendet, wenn der Basic-Befehl „CLS“ ausgeführt wird. Dadurch kann man das ganze Ausgabefenster mit einem eigenen Muster ausfüllen (siehe **Abb. 4**).

### 7.2.7. Flächen zeichnen

Es gibt verschiedene Flächen, die man mit den ROM-Routinen zeichnen kann:

**Rect** ist ein ganz normales Rechteck. Zur Beschreibung dieser Fläche genügt ein Parameter vom Typ „Rect“ (s.o.).

**Oval** ist eine Ellipse. Sie wird durch ein Rechteck definiert, dessen Höhe und Breite gerade die Höhe und Breite der Ellipse angeben.

Zur Beschreibung dieser Fläche genügt also ebenfalls ein Parameter vom Typ „Rect“.

**RoundRect** ist ein Rechteck, bei dem die Ecken abgerundet sind (z.B. der gesamte Schreibtisch des Mac). Die runden Begrenzungslinien in den Ecken sind je ein Viertel einer Ellipse.

Um diese Fläche zu beschreiben reicht ein Parameter vom Typ „Rect“ nicht aus. Zusätzlich benötigt man noch die Breite und die Höhe der Ellipse, die die abgerundeten Ecken definieren.

**Arc** ist ein Ausschnitt einer Ellipse, der durch zwei Winkel gegeben ist (aber anders als bei „CIRCLE“!).

Man benötigt zur Beschreibung dieser Fläche einen Parameter vom Typ „Rect“ (definiert die Ellipse) und die zwei Winkel.

Diese Winkel sind hier im Gradmaß angegeben. Ein Winkel von 0 Grad zeigt nach oben, einer von 90 Grad nach rechts. Der erste Winkel ist der Anfangswinkel, während der zweite Winkel angibt, wie weit und in welche Richtung vom Anfangswinkel aus gezeichnet werden soll. Beispiele: 90 Grad, 180 Grad – zeichnet den Ausschnitt von 90 bis 270 Grad.

90 Grad, -180 Grad – zeichnet den Ausschnitt von 90 bis -90 Grad.

Jede dieser 4 Flächen kann nun auf verschiedene Arten gezeichnet werden:

**Frame** – die Fläche wird mit dem „Pen“ umrahmt. Dabei kommen alle Eigenschaften des Pen zum Tragen, also die Größe, das Muster, usw.

Der Pen wird dabei so geschickt geführt, daß nur *innerhalb* der angegebenen Fläche gezeichnet wird. Dies ist deshalb nicht trivial, weil der Pen ja ein Rechteck mit veränderlicher Größe ist.

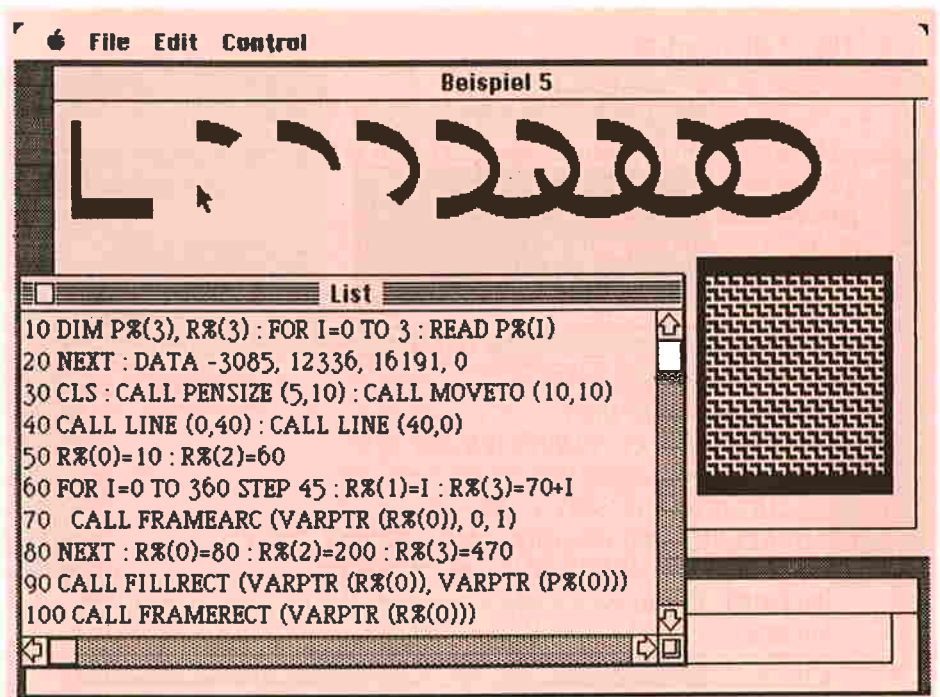


Abb. 5

**Paint** – die Fläche wird in der sogenannten „Vordergrundfarbe“ gezeichnet (normalerweise in schwarz).

**Erase** – zeichnet die Fläche in der „Hintergrundfarbe“, normalerweise in weiß. Dadurch wird diese Fläche praktisch gelöscht.

**Invert** – wandelt alle Pixel in der Fläche in das logische Gegenteil um: aus weiß wird schwarz und umgekehrt.

**Fill** – füllt die Fläche mit einem bestimmten Muster (Pattern, s.o.). Dieses Muster muß man bei dem Befehl zusätzlich angeben.

Für jede der 4 Flächen stehen also 5 verschiedene Zeichenmöglichkeiten zur Verfügung, was eine Zahl von 20 verschiedenen Routinen ergibt.

Die Namen dieser Routinen setzen sich wie folgt zusammen: Das erste Teilwort gibt die Zeichenart an (z.B. „Erase“), das zweite Teilwort die Form der Fläche (z.B. „Oval“). Der Name dieser Routine lautet dann „EraseOval“.

Beispiele zu den möglichen Aufrufen mit Parametern sieht man in den **Abb.4 und 5** und in den nachfolgenden Beispielen:

```

CALL PaintRect (VARPTR (R%(0)))
CALL FillOval (VARPTR (R%(0)), VARPTR (P%(0)))
CALL InvertRoundRect (VARPTR (R%(0)), B, H)
CALL EraseArc (VARPTR (R%(0)), A, W)
CALL FrameRect (VARPTR (R%(0)))

```

Dabei sei R% ein Array mit 4 Elementen (Rect), P% ebenfalls ein Array mit 4 Elementen (Pattern), B und H die Breite und Höhe einer Ellipse, A und W zwei Winkel.

## Computer-unterstütztes Lernen für Beruf, Weiterbildung, Schule



INTUS SOFTWARE

Kaiserstraße 21, 7890 Waldshut, Telefon 0 77 51 / 79 20

- Programme für Apple IIe/c/teilw. +.
- Demo-Diskette mit 8 Programmen und 7 Denkspielen DM 10,-
- Gesamtkatalog mit über 160 Titeln kostenlos.

**NEU:** Rücknahme von Lernprogrammen (wenn Sie den Stoff intus haben) zu 50 % des Kaufpreises bei Kauf eines anderen intus-Lernprogrammes.



# Grafik-Demonstrationen

von Ralf Knoke

Die abgedruckte Programmsammlung **GRAFIK.DEMOS** soll ein Beispiel dafür sein, wie man auch als Anfänger mit kurzen Programmschritten schöne Grafiken im HGR-Bereich auf dem Apple II erzeugen kann. Sie soll zur Nachahmung anregen. Es werden verschiedene Grafiken, sowohl zwei- wie auch dreidimensional, gezeigt. Im Listing sind die einzelnen Teile durch entsprechende REMs voneinander getrennt.

Vor den eigentlichen Zeichenanweisungen (HPLOTs) werden meistens die Koordinaten der Endpunkte festgelegt. Dies erscheint immer dann sinnvoll, wenn Punkte mehrmals angesteuert werden müssen. Das war allerdings nicht überall der Fall, weshalb manche Punkte direkt in der HPLOT-Anweisung angegeben werden. Hierzu ist für alle, die es selbst versuchen wollen, noch zu sagen, daß die Punkte

natürlich nicht willkürlich gewählt worden sind. Man muß ihre Lage mit dem Computer ausprobieren oder eine Skizze auf einem karierten DIN-A4-Blatt anfertigen. Eine Kästchenlinie entspricht dabei 5 HPLOT-Punkten. Das hört sich schwieriger an, als es ist. Nach einiger Zeit hat man jedoch „den Bogen raus“.

Arbeitet man im 80-Zeichenmodus, so sollte man in Zeile 1720 die Variable X auf 6 vergrößern und in den HTAB-Anweisungen die addierten Zahlen verdoppeln, z.B. 5 auf 10, 30 auf 60 usw. Dadurch wird die Beschriftung eindeutiger.

Mit den Schleifen, die z.B. in den Zeilen 1850–1920 vorkommen, erreicht man, daß Linien im 3-D-Bereich, die eigentlich nicht sichtbar sind, nur unterbrochen dargestellt werden. Dies ist einfach, wenn die Linien waagrecht, senkrecht oder im 45 Grad Winkel stehen, sonst gibt es Schwierigkei-

ten, da der Computer die Linien bricht. Beim N-Eck und bei der Tortenstück-Grafik wurden Eck- bzw. Kreisformeln notwendig. In einem kurzen Artikel ist es schwierig, ihre Arbeitsweise Anfängern einfach und deutlich zu erklären. Ein Tip: Experimentieren Sie etwas, dann wird Ihnen alles klar. Denn das Programm soll ja zum Experimentieren anregen. Für Fortgeschrittene sind die Formeln leichter zu durchschauen, zumal sie wohl auch öfter eingesetzt werden. Abschließend ist zu sagen, daß die Programme einen Mittelweg zwischen Übersichtlichkeit und Kürze beschreiten. Aus diesem Grunde wurde auch bewußt auf ein Menü zur Auswahl der einzelnen Figuren verzichtet. Wer dies nicht missen möchte, wird das Problem auch als Anfänger durch Eingabe eines Strings und GOTO bzw. GOSUB lösen können.

Und nun viel Spaß!

## Applepreise = Mondpreise?

Die sog. **Preisbindung** der zweiten Hand (= vertikale Preisbindung = Festlegung der für den Einzelhandel verbindlichen Endabnehmerpreise durch den Produzenten) war früher für Markenartikel schlechthin zulässig. Seit 1974 können nur noch Verlagerzeugnisse der vertikalen Preisbindung unterworfen werden, während für Markenartikel bestenfalls die sog. unverbindliche **Preisempfehlung** zulässig ist. Als **Mondpreis** definiert man den „Mißbrauch von vertikalen Preisempfehlungen, indem die empfohlenen Bruttopreise zu hoch festgesetzt werden, um dem Einzelhandel die Unterbietung auch bei üblicher Kalkulation zu ermöglichen und damit dem Kunden besondere Preiswürdigkeit vorzutäuschen“ („Gablers Wirtschaftslexikon“). Die unverbindliche Preisempfehlung muß „in der Erwartung ausgesprochen werden, daß der empfohlene Preis dem von der Mehrheit der Empfehlungsempfänger voraussichtlich geforderten Preis entspricht“ (§ 38a GWB). Das Bundeskartellamt ist in diesem Punkt nicht nur ungewöhnlich penibel, sondern hinsichtlich der Geldbußen auch ausgesprochen drakonisch. Vor diesem Hintergrund muß man sich fragen, ob

die unverbindlichen Preisempfehlungen für Apple-Produkte Mondpreise sind, denn für eine Reihe von Produkten ist der „Marktpreis“ (= vom Einzelhändler üblicherweise geforderter Preis) oft bis zu oder teils sogar über 50% niedriger als der empfohlene Preis. Einige Beispiele (aufgrund der Apple-Preisliste, Stand März 1985):

Apple IIe Grundgerät: empfohlen DM 3400,-, dagegen z.B. bei Vobis DM 1998,-  
Apple IIc Grundgerät: empfohlen DM 3750,-, dagegen z.B. bei ProSoft DM 2649,-

512K-Erweiterung für Macintosh: empfohlen DM 3780,- dagegen z.B. bei Schapach DM 1680,- oder bei Hoco DM 1690,-  
Macintosh Grundgerät mit 512K: empfohlen DM 12250,-, dagegen z.B. bei Koslik DM 9500,-

Die Vergleichspreise sind Anzeigen entnommen. Ergiebiger ist die Auswertung der tatsächlichen Preislisten der Einzelhändler.

Für mich als Redakteur sind diese Preisabweichungen – gelinde gesagt – ein Ärgernis. Als ich den Testbericht zu der AP33

(= 1024K-RAM-Karte) der Firma IBS erstellte, die im Handel für ca. DM 3400,- (mit geringfügigen Schwankungen nach oben und unten) erhältlich ist, bot sich ein Preisvergleich zu der 512K-Mac-Erweiterung an. Welchen Preis sollte ich jedoch hier ansetzen, den „empfohlenen“ Händlerpreis von DM 3780,- oder den um über 50% niedrigeren „üblichen“ Händlerpreis von ca. DM 1700,-? Wenn Schulen und Universitäten Rabatte erhalten, dürften wohl die empfohlenen Preise und nicht die Marktpreise als Bezugsbasis dienen. Jedenfalls fiel es dem „Empfehlungsempfänger“ dann nicht schwer, ca. 20-30% abzuziehen, denn damit läge er immer noch über dem Marktpreis. Also nur eine Augenwischerei?

Ich kann hier nur hoffen, daß man sich zu einer realistischeren Preispolitik entschließt oder Listen mit „unverb. empf. Verkaufspreisen incl. MwSt.“ gar nicht mehr an Endabnehmer abgibt, denn schon manche „empfohlene“ Augenwischerei ist ins Auge gegangen.

U. Stiehl



## GRAFIK.DEMOS

```
1000 REM Ralf Knoke
1010 REM Oktober 1984
1020 :
1100 REM Raster-Muster
1110 HOME : HGR2 : HGR : M = 7
1120 HCOLOR= M
1130 FOR I = 0 TO 279 STEP 3: HPLLOT I,0 TO I,191: NEXT
1140 FOR I = 0 TO 159 STEP 3: HPLLOT 1,I TO 279,I: NEXT
1150 IF M = 7 THEN M = 0: GOTO 1120
1160 GET A$: HOME
1170 HGR2
1180 :
1200 REM Fünfeck-Stern
1210 HCOLOR= 7
1220 HPLLOT 85,15 TO 140,180 TO 195,15 TO 52,117 TO 228,117
    TO 85,15
1230 GET A$: HGR2 : HCOLOR= 3
1240 :
1300 REM Keplerscher Sternkörper
1310 A = 140:B = 95:C = 5:D = 227:E = 67:F = 193:G = 167:H
    = 87:I = 53:J = 23:K = 123:L = 185:M = 76:N = 106
1320 HPLLOT A,B TO A,C
1330 HPLLOT A,B TO D,E
1340 HPLLOT A,B TO F,G
1350 HPLLOT A,B TO H,G
1360 HPLLOT A,B TO I,E
1370 HPLLOT F,J TO I,K TO D,K TO H,J TO A,L TO F,J
1380 HPLLOT F,J TO A,39 TO H,J TO H,M TO I,K TO N,A TO A,L
    TO 176,A TO D,K TO F,M TO F,J
1390 HPLLOT 120,157 TO H,G TO H,133: HPLLOT 161,158 TO F,G TO
    F,134
1400 HPLLOT 207,93 TO D,E TO F,57: HPLLOT 160,33 TO 140,5 TO
    120,33: HPLLOT 87,57 TO I,E TO 73,93
1410 :
1500 REM Stern-Löscher
1510 GET A$: HCOLOR= 7
1520 FOR I = 0 TO 140: HPLLOT I,0 TO I,191: HPLLOT 279 - I,0
    TO 279 - I,191: NEXT
1530 GET A$
1540 :
1600 REM Pythagoras
1610 HGR2 : HCOLOR= 3
1620 A = 156:B = 169:C = 109:D = 96:E = 124:F = 70
1630 HPLLOT B,D TO B,A TO C,A TO C,D TO B,D TO E,F TO C,D TO
    84,81 TO 99,55 TO E,F TO 151,25 TO 196,51 TO B,D
1640 GET A$
1650 :
1700 REM Farbtafel
1710 HOME : HGR
1720 X = 3:F = 0
1730 VTAB (22): PRINT "FARBE : ";
1740 VTAB (24): HTAB (X): PRINT F;: HTAB (X + 5): PRINT F +
    1;: HTAB (X + 10): PRINT F + 2;: HTAB (X + 15): PRINT
    F + 3;: HTAB (X + 20): PRINT F + 4;: HTAB (X + 25):
    PRINT F + 5;: HTAB (X + 30): PRINT F + 6;: HTAB (X +
    35): PRINT F + 7;
1750 M = 0:D = 1
1760 HCOLOR= M
1770 FOR I = 0 TO 279: HPLLOT I,0 TO I,159
1780 IF I = D * 35 THEN D = D + 1:M = M + 1: HCOLOR= M
1790 NEXT
1799 :
1800 REM 3-D-Quader
1810 GET A$: HOME : HGR : HGR2
1820 HCOLOR= 7: HPLLOT 65,85 TO 155,85 TO 155,175 TO 65,175
    TO 65,85 TO 128,21 TO 218,21 TO 218,111 TO 155,175
1830 HPLLOT 155,85 TO 218,21
1840 A = 128:B = 111
1850 FOR I = 1 TO 22
1860 HPLLOT A,B TO A + 2,B:A = A + 4: NEXT
1870 A = 128:B = 21
1880 FOR I = 1 TO 22
1890 HPLLOT A,B TO A + 2,B: B = B + 4: NEXT
1900 A = 128:B = 111
1910 FOR I = 1 TO 16
1920 HPLLOT A,B TO A - 2,B + 2:A = A - 4:B = B + 4: NEXT
1930 :
2000 REM Pyramide
2010 GET A$: HGR2 : HCOLOR= 7
2020 HPLLOT 126,157 TO 149,36 TO 83,114 TO 126,157 TO
    216,157 TO 149,36 TO 173,114
2030 A = 83:B = 114
2040 FOR I = 1 TO 22: HPLLOT A,B TO A + 2,B:A = A + 4
2050 NEXT
```

```
2060 A = 216:B = 157
2070 FOR I = 1 TO 11: HPLLOT A,B TO A - 2,B - 2:A = A - 4:B
    = B - 4: NEXT
2080 :
2100 REM N-Eck
2110 GET A$
2120 Z = 1
2130 N = INT (( RND (1) * 120))
2140 HGR2
2150 HCOLOR= 3
2160 PI = 4 * ATN (1):Q = 2 * PI / N
2170 A = 220:B = 96
2180 C = A:D = B
2190 G = 5
2200 FOR I = 1 TO N
2210 X = 80 * COS (Q * I) + 140:Y = 80 * SIN (Q * I) + 96
2220 HPLLOT X,Y TO 140,96: HPLLOT X,Y TO C,D:C = X:D = Y
2230 NEXT
2240 REM Wird in Zeile 2260 "Z" vergrößert,
2250 REM dann finden mehr Durchläufe statt.
2260 Z = Z + 1: IF Z = 3 THEN GOTO 2300
2270 GET A$: GOTO 2130
2280 :
2300 REM Tortenstück-Grafik
2310 GET A$
2320 PI = 3.14159265357
2330 DEF FN DR(W) = (W * PI) / 180
2340 NX = 140
2350 NY = 96
2360 RA = 95
2370 HGR2 : HCOLOR= 7
2380 GOSUB 3000
2390 RA = 95 / 2: GOSUB 3000
2400 FOR I = 1 TO 3
2410 WB = FN DR(I * 120)
2420 X = RA * COS (WB):Y = RA * SIN (WB)
2430 X = NX + X:Y = NY + Y
2440 HPLLOT NX,NY TO X,Y
2450 NEXT
2460 FOR I = 1 TO 3
2470 X1 = NX + 95 * COS ( FN DR(I * 120 + 60)):Y1 = NY + 96
    * SIN ( FN DR(I * 120 + 60))
2480 X2 = NX + 95 / 2 * COS ( FN DR(I * 120 + 60)):Y2 = NY
    + 95 / 2 * SIN ( FN DR(I * 120 + 60))
2490 HPLLOT X1,Y1 TO X2,Y2
2500 NEXT
2510 GET A$: HOME : TEXT : END
2520 :
3000 FOR I = 0 TO 360
3010 WB = FN DR(I)
3020 X = RA * COS (WB):Y = RA * SIN (WB)
3030 X = NX + X:Y = NY + Y
3040 IF I = 0 THEN HPLLOT X,Y: GOTO 3060
3050 HPLLOT X1,Y1 TO X,Y
3060 X1 = X:Y1 = Y
3070 NEXT
3080 RETURN
```



## Diversi-DOS 2-C

Die 4-C-Dateien HELLO und ASMDIV ändern ein im Speicher befindliches Original-DOS-3.3 in Diversi-DOS 4-C um. Trotz gleicher Dokumentation gilt dies nicht für die Dateien HELLO und ASMDIV von Version 2-C, die auf der Peeker-Sammeldisk # 6 enthalten sind (s. Peeker, Heft 6/85, S. 74). Der Einfachheit halber haben wir deshalb zusätzlich die 4-C-Dateien auf die Sammeldisk # 8 aufgenommen. Die Implementierung unter 4-C ist übrigens noch einfacher: Original-DOS-3.3 booten, dann Sammeldisk # 8 einlegen und HELLO starten (RUN HELLO). Wenn das Menü erscheint, ist DOS 3.3 bereits gepatcht. Hauptmenü verlassen und mit INIT Leerdiskette formatieren. Dies ist alles.

Inzwischen ist die Firma DSR umgezogen. Die neue Anschrift lautet:

Diversified Software Research, 34880 Bunker Hill, Farmington, MI 48018-2728, USA.

# Zeichenjagd

## Ein Einzelier

von Hans-Peter Lendle

ZEICHENJAGD ist ein kleines Spiel und Trainingsprogramm mit der Tastatur des Computers. Ein einzelner Buchstabe erscheint irgendwo auf dem Bildschirm und muß innerhalb einer bestimmten Zeit mit der entsprechenden Taste „erwischt“ werden. Ein Tonsignal belohnt den Erfolg, und die Wartezeit wird um eine Stufe verkürzt. Erreicht man Stufe 2, so kommt der nächste Buchstabe im Alphabet dazu, ebenso wenn man von Stufe 1 auf 2 abfällt (damit der nächste Aufstieg beschwerlicher wird).

### Spielverlauf

Tippt man keine oder die falsche Taste, so wird die Wartezeit um eine Stufe verlängert. Fällt man auf Stufe 9 zurück, wird ein Buchstabe weggenommen (zur Erleichterung). Vier Buchstaben bleiben jedoch in jedem Fall übrig.

Das Programm startet mit Stufe 0 und Buchstabe „A“. Es „bremst“ sich, wenn keine Eingabe erfolgt, auf Stufe 9 ab. Diese Stufe ist bequem für das „Ein-Finger-Adler-Suchsystem“ geeignet. Wird „A“ eingetippt, so erhöht sich die Spielstufe auf 2, bei der der Buchstabe „B“ hinzukommt. Mit etwas Geschick gesellen sich auch noch „C“ und „D“ dazu, mit sinkender Wahrscheinlichkeit weitere Buchstaben – man fällt zurück.

Jetzt muß man sich Buchstabe um Buchstabe und immer wieder Stufe um Stufe hocharbeiten. Stufe 1 mit allen Buchstaben des Alphabets ist wohl sogar für „Zehn-Finger-Profis“ auf die Dauer schwer zu halten.

Der aktuelle Leistungsstand steht oben in der Mitte des Bildschirms: die Spielstufe invers als Ziffer von 1 bis 9, der letzte zu erwartende Buchstabe blinkend daneben.

### Programmbeschreibung

Die Beschreibung zeigt, daß das Programm durchaus nicht linear abläuft. Wenn man alles in einer Zeile unterbringen will, braucht man zwangsläufig etwas, was IF-THEN-ELSE in Applesoft simuliert. Wie

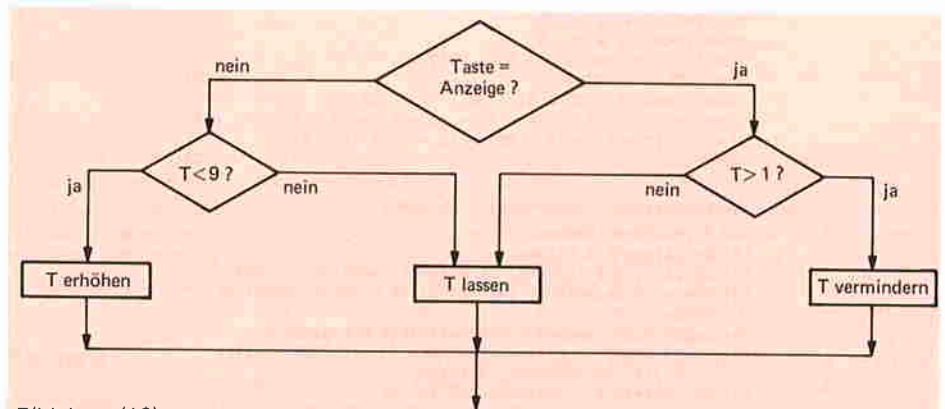


Bild 1 zu (10)

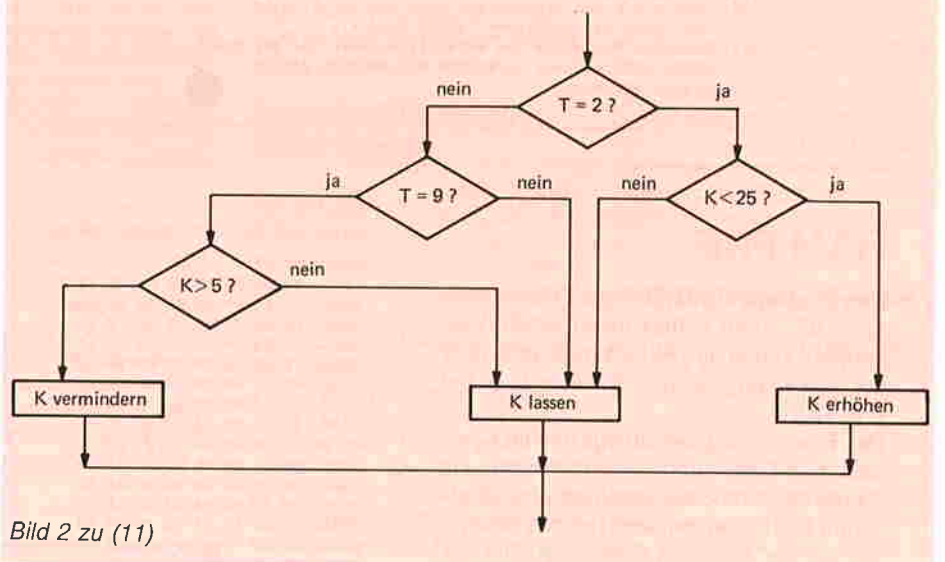


Bild 2 zu (11)

von Franz-Josef Hüskens in Peeker 2/84 beschrieben, geht das mit Verknüpfungen von Wahrheitswerten.

Bei der ZEICHENJAGD geschieht dies in den Teilen (9) bis (11). Dort nehmen die logischen Ausdrücke ( $P = B$ ) oder ( $P <> B$ ) die Werte 1 oder 0 an, die dann in arithmetische Ausdrücke eingebaut werden können. **Bild 1 und 2** stellt diese umfangreichen Teile noch einmal als Flußdiagramm dar.

Das Programm selbst besteht aus 13 Anweisungen (das NEXT im Abschnitt (7) mitgezählt), die in einer einzigen Zeile untergebracht sind – und doch ist es ein durchaus brauchbares Trainingsprogramm für Anfänger und Könner (und mindestens

so spannend wie die ersten Computer-Pingpongspiele).

Beim Eingeben des Programms sollten keine Leerzeichen getippt werden, weil sonst die Eingabezeile zu lang werden könnte.

Wer auf bestimmte Buchstaben oder andere Zeiten mehr Wert legt als auf eine Verkürzung, kann sich ohne großen Aufwand das individuelle Maßprogramm aus mehreren Applesoft-Zeilen neu schreiben. Meine böse BASIC-Fanatiker-Seele kann sich die Frage an die Pascal-Enthusiasten nicht verkneifen: „Geht so 'was auch in Pascal?“ Aber ganz gleich, wo Sie stehen, ich wünsche Ihnen viel Spaß mit der ZEICHENJAGD, sei es beim Spielen, beim Trainieren oder beim Ändern.

## ZEICHENJAGD (Einzeiler)

Die nachgestellten Zahlen in Klammern sind nicht einzugeben.

```
1 HOME : (1)
Z = RND (1): (2)
POKE 1040,T + 48: (3)
POKE 1044,K + 64: (4)
B = INT (Z * K) + 193: (5)
POKE 1244 + Z * 724,B: (6)
FOR I = 1 TO 350 * T - 250: NEXT : (7)
P = PEEK ( - 16384): (8)
PRINT CHR$( 7 * (P = B) + 32 * (P < > B));: (9)
T = T + (B < > P) * (T < 9) - (B = P) * (T > 1): (10)
K = K + (T = 2) * (K < 25) - (T = 9) * (K > 5): (11)
GOTO 1 (12)
```

Erklärung zu den einzelnen Abschnitten:

- (1) Bildschirm löschen;
- (2) Zufallszahl Z bilden;
- (3) Spielstufe T (invers) in Bildschirmspeicher POKEn;
- (4) Höchsten Buchstaben (blinkend) in Bildschirmspeicher POKEn;
- (5) neuen Buchstaben B aus Zufallszahl Z bestimmen;
- (6) Buchstabe B in Bildschirm POKEn (diese POKEs können auch die "Screenholes" treffen);
- (7) der Spielstufe entsprechend warten;
- (8) eingegebenen Buchstaben P aus Tastaturspeicher lesen;
- (9) Tonsignal geben, wenn Buchstabe stimmt;
- (10) Spielstufe dem derzeitigen Wert und der Eingabe entsprechend neu bestimmen (siehe Bild 1);
- (11) höchsten Buchstabe dem derzeitigen Wert und der Spielstufe entsprechend verändern (siehe Bild 2);
- (12) wieder von vorn.



## RAM.FRE

Das Programm RAM.FRE aus Peeker, Heft 1/2 - 85, S. 40 enthält einen schwerwiegenden Fehler im Algorithmus und läuft nur in wenigen Fällen einwandfrei (z.B. bei FRE.TEST).

Das Problem liegt bei Strings, die im Speicher eine Seitengrenze überschreiten. Da jeweils nur eine Seite gepuffert wird, übernimmt das Programm den Teil des Strings, der auf der nächsten Seite liegt, aus der Page 3 (der Puffer umspannt den Bereich \$0200 bis \$02FF). Die somit übertragene Zeichenkette enthält dann im hinteren Teil nur noch Schrott (um genau zu sein: einen Teil der Treiberoutine).

In der neuen Version RAM.FRE.NEU werden die Strings nicht mehr im Eingabepuffer gesichert, sondern in den zwei Seiten von \$9400 bis \$95FF, wobei jedoch nur die Strings, die im Bereich \$9400 bis \$94FF liegen, bearbeitet werden können.

**Diese Probleme treten bei der üblicherweise benutzten LC-Version nicht auf.**

Aus Platzgründen kann hier nur der Hexdump abgedruckt werden, der Quelltext ist auf der Peeker-Sammeldiskette enthalten.

H. Grumser

RAM.FRE.NEU

BSAVE RAM\_FRE.NEU, A\$92B6, L\$147

```
92B6- A0 10
92B8- B9 D2 92 99 00 03 88 10
92C0- F7 A9 FF 85 E3 A9 92 A0
92C8- E3 85 74 B4 73 85 70 84
92D0- 6F 60 38 A5 6F E5 6D A5
92D8- 70 E5 6E C5 E3 B0 03 4C
92E0- E3 92 60 A5 70 8D FF 93
92E8- A5 73 85 6F 18 F0 01 38
92F0- A5 74 85 70 E9 00 8D FD
92F8- 93 38 A5 69 E9 07 85 3E
9300- A5 6A E9 00 85 3F A5 6B
9308- 8D FE 93 20 B4 93 A6 6C
9310- 20 22 93 20 52 93 CE FD
9318- 93 AD FD 93 CD FF 93 B0
9320- D9 60 18 A5 3E 69 07 85
9328- 3E 90 02 E6 3F 45 6B D0
9330- 04 E4 3F F0 EC A0 00 B1
9338- 3E C8 51 3E 10 E4 B1 3E
9340- 10 E0 A0 04 B1 3E CD FD
9348- 93 90 D8 D0 D5 20 D0 93
9350- F0 D0 20 7B 93 B0 5C A0
9358- 02 B1 3E CD FD 93 90 05
9360- D0 03 20 D0 93 18 A9 03
9368- 65 3E 85 3E 90 02 E6 3F
9370- CD FE 93 D0 E2 E4 3F D0
9378- DE F0 D7 18 AD FE 93 85
9380- 3E 86 3F 45 6D 0F 04 E4
9388- 6E F0 28 A0 02 B1 3E 65
9390- 3E 8D FE 93 C8 B1 3E 65
9398- 3F AA A0 00 B1 3E C8 51
93A0- 3E 10 D8 A0 04 B1 3E 0A
93A8- 69 05 65 3E 85 3E 90 03
93B0- E6 3F 18 60 A0 00 AD FD
93B8- 93 84 3A 85 3B B1 3A 99
93C0- 00 94 C8 D0 F8 E6 3B B1
93C8- 3A 99 00 95 C8 D0 F8 60
93D0- A9 94 85 3B 88 B1 3E 85
93D8- 3A 88 38 A5 6F F1 3E 85
93E0- 6F C8 91 3E A5 70 E9 00
93E8- 85 70 C8 91 3E 88 88 B1
93F0- 3E F0 09 A8 88 B1 3A 91
93F8- 6F 98 D0 F8 60
```

## DB-MEISTER

### Adreß- und Schemabriefprogramm

Der DB-Meister ist ein in Assembler geschriebenes, ungewöhnlich schnelles, unkompliziertes und zugleich „narrensicheres“ Adreß-, Datei- und Schemabriefprogramm.

Der DB-Meister dient zum Anlegen, Pflegen, Sortieren, Selektieren und Ausdrucken von Dateien aller Art. Als Apple-Benutzer wissen Sie, wie langsam viele Programme dieser Art sind. Nicht so der DB-Meister!

Drei Beispiele:

- Jeder beliebige von 560–999 Records wird nach Indexfeldern in 0,2 Sekunden gefunden.
- Eine komplette Datendiskette mit z. B. 600 Records läßt sich in 1 Minute nach 3 Feldern sortieren und untersortieren. Dabei ist die Zeit für Diskettenzugriff bereits mitgerechnet.
- Das Einlesen eines 50 Sektoren langen Programm-Moduls dauert nur 3,5 Sekunden.

Technische Daten des DB-Meisters

- Recordlänge bis zu 230 Zeichen
- 560 bis 1000 Records pro Datendiskette
- Maximal 25 Felder pro Record
- 4 Datentypen (String, Integer, Dezimalzahl, Real)
- Suche nach 3 Indexfeldern – je 4 Zeichen lang – mit Wildcard-Funktion
- Sortieren und Filtern (kumuliertes Selektieren) geschieht nach den Index-Feldern
- Ausdruck der Dateien als Etiketten, Listen und Schemabriefe (mit Felder- und Tastatureinschieben an beliebigen Stellen des Formbriefes)
- normal kopierbare Programmdiskette, unterteilt in Hauptprogramme und diverse Hilfsprogramme
- einsatzfähig auf Apple IIe oder IIc. (Achtung: Brief-Modul läuft nicht mit Videx-Karte!)
- 256K RAM-Disks verwendbar

**Gesamtpreis 290,- (2 Disketten + gedrucktes Manual)**

U. Stiehl

c/o Dr. A. Hühlig Verlag  
Postfach 10 28 69 · 6900 Heidelberg



# Die Polaroid-Foto-Systeme

getestet von **Thomas Bühner**  
und **Prof. Dr. Klaus Hausmann**

Zur Foto-Dokumentation von Computer-Grafiken griff man bisher zur vertrauten Kleinbildkamera. Polaroid hat jetzt zwei Systeme entwickelt, die bessere Ergebnisse erwarten lassen.

Wer häufiger darauf angewiesen ist, Diagramme, Schaltpläne oder andere mit Computer-Hilfe geschaffene Grafiken zu Papier zu bringen, hatte bisher zwei generelle Möglichkeiten zur Dokumentation seiner Arbeit: einerseits die Ausgabe des gespeicherten Bildes mittels spezieller Hardcopy-Geräte – üblicherweise Plotter oder Matrixdrucker – auf Zeichenpapier und andererseits die Fotografie vom Schwarzweiß- oder Farb-Monitor.

## Plotter und Matrixdrucker

Der Einsatz des Plotters ist oft nicht möglich, da er nur von einer beschränkten Anzahl von Grafikprogrammen angesprochen werden kann. Matrixdrucker liefern zwar eine punktgenaue Wiedergabe des Originals, verzerren aber auf Grund unterschiedlicher hori-

zontaler und vertikaler Dehnungsfaktoren oft Kreise zu Ellipsen und Quadrate zu Rechtecken. Da mehrfarbige Matrixdrucker noch wenig verbreitet sind, entfällt auch die Möglichkeit zu bunten Reproduktionen.

## Normale Kamera

Die Frage Schwarzweiß oder Bunt spielt keine Rolle bei der Fotografie vom Bildschirm. Für beide Fälle gibt es geeignetes Material im Fachhandel. Ein Foto ist außerdem die originalgetreueste Form der Bild-Dokumentation. Dennoch wird dieser Weg selten eingeschlagen, da die Nachteile zu schwer wiegen. Denn wie geht man vor? Zunächst wird die Kamera auf ein Stativ montiert und vor dem Monitor aufgebaut. Dabei muß man genau darauf achten, daß die Filmebene parallel zur Ebene des Bildschirms liegt, da sonst eine verzerrte Aufnahme die Folge ist. Helligkeit und Kontrast oder Farbsättigung des Bildes müssen genau stimmen. Dann wird der Raum abdunkelt, um störende Reflexe zu

vermeiden. Um wenigstens ein oder zwei verwertbare Negative oder Dias zu erhalten, macht man nun eine Serie von Aufnahmen mit unterschiedlicher Zeit-/Blenden-Kombination. Schließlich gibt man den Film ins Labor und erhält einige Tage später die fertigen Abzüge. Doch selbst dann, wenn man eine eigene Dunkelkammer besitzt, wird man bestenfalls nach ein paar Stunden die Ergebnisse vorliegen haben. Und diese Prozedur muß für jede Foto-Dokumentation erneut durchlaufen werden.

## Polaroid-CU-5-Makro-Modul-System

Im Vergleich zur herkömmlichen Bildschirm-Fotografie ist der Einsatz des CU-5-Makro-Modul-Systems kinderleicht. Aus den vielfältigen Kombinationsmöglichkeiten der verfügbaren Module soll hier nur die gezeigt werden, mit der man Sofortbilder vom 12-Zoll-Monitor macht. Für diesen Zweck besteht die Kamera aus dem Kamerakörper, dem Linsensystem und einem schwarzen Kunststofftrichter. Der Film wird eingelegt, Belichtungszeit und Blende nach den Empfehlungen des Herstellers eingestellt und der Trichter an den Bildschirm gepreßt. Dann betätigt man den Abzug – ähnlich wie bei einer Pistole –, entnimmt die Film-/Entwickler-Folie, wartet eine Minute und zieht anschließend das fertige Foto ab.

Das Scharfstellen entfällt, da die Schärfenebene des Linsensystems genau auf dem Bildschirm liegt. Ein Abdunkeln des Raums wird ebenso überflüssig, denn alle Lichtquellen sind durch den Kunststofftrichter abgeschirmt. Vor der ersten Benutzung des Geräts ist es allerdings ratsam, das Innere mit einem schwarzen Mattlack zu bestreichen, da das Kunststoffmaterial selbst so stark reflektiert, daß unter Umständen Spiegelungen der einzigen Lichtquelle, die nicht abschirmbar ist, auftreten können: die des Monitors selbst.

Durch die gute Abbildungsqualität fällt nun ein Manko ins Auge: Da sich der Elektronenstrahl, der den auf der Röhre aufgetragenen Leuchtstoff zum Fluoreszieren bringt, horizontal über das Bild bewegt, entstehen Rasterlinien, die bei der Betrachtung des Fotos nun noch stärker ins Auge fallen als zuvor am Monitor selbst. Bei Farbfotos muß man leider auch feststellen, daß das empfohlene Filmmaterial nicht für diesen Zweck geeignet ist. Die Farben wirken insgesamt sehr blaß, und der Rot-Anteil fehlt auf dem Foto fast ganz. Für Schwarzweiß-Aufnahmen stellt das Polaroid-CU-5-Makro-System aber eindeutig ein leicht zu handhabendes Mittel dar, dessen Ergebnisse einfachen bis mittleren Ansprüchen genügen.

## Polaroid-Palette-System

Bei der Entwicklung des Palette-Systems besann man sich darauf, wie die Farberzeugung beim Fernsehgerät vor sich geht: Die drei Grundfarben Rot, Grün und Blau werden auf dem Bildschirm in unterschiedlichen Anteilen übereinander projiziert und vermischen sich so für unser Auge. Hätten wir ein fotografisches Gedächtnis, so könnten auch zuerst alle Rot-Anteile des Bildes gezeigt werden, anschließend alle Partien, die Grün enthalten, und zum Schluß das Blau. Nun, für einen menschlichen Beobachter wäre eine solche Bild-Betrachtung ungeeignet, eine Fotokamera jedoch könnte keinen Unterschied wahrnehmen zwischen einer gleichzeitigen und einer aufeinanderfolgenden Darbietung der drei Grundfarben.

So besteht also das Palette-System aus einem beigen Metallgehäuse, das ein Netzteil, einen kleinen Schwarzweiß-Bildschirm, ein Rad mit Filtern in den drei Grundfarben und einen Motor enthält, der das Rad dreht. An der Rückseite befinden sich ein RS-232-C-Eingang, Monitor-Ein- und -Ausgang, ein Anschluß für einen Foto-Win-

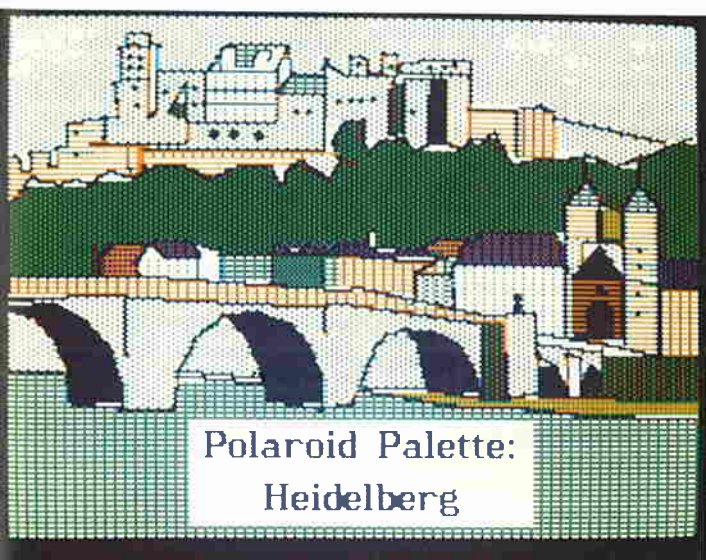


Bild 1

der und die bei Monitoren üblichen Regler.

Nimmt man das Gerät zum ersten Mal in Betrieb, so wird zunächst über das mitgelieferte Kabel eine Verbindung zum RS-232-C-Interface, das in einem der Apple-Slots steckt, hergestellt. Dieses Interface ist nicht im Lieferumfang enthalten; viele Anwender werden vermutlich die Super-Serial-Card von Apple verwenden. Dann schließt man den Monitor des Palette-Systems am Computer an. Nun wird die Programm-Diskette ins Laufwerk gelegt und gestartet. Das Gerät muß nun noch justiert werden, was in weniger als fünf Minuten erledigt ist. Jetzt ist es betriebsbereit. Beim nächsten Mal entfallen alle diese Schritte, lediglich eine Aufwärmzeit von 20 Minuten muß eingehalten werden. Je nach Wunsch befestigt man die Sofortbild- oder die 35-mm-Kamera an der Vorderseite.

Alle Aufnahmen zu diesem Artikel entstanden, soweit nicht anders vermerkt, mit der Sofortbild-Kamera auf Polaroid Polacolor ER Land Pack Film Type 669.

Besitzt man nur ein Diskettenlaufwerk, muß man bei jeder neuen Aufnahme einen Disk-Wechsel vornehmen. Bei zwei Drives ist das nicht nötig. Die Bilder müssen als Binär-File auf Diskette vorliegen. Zu Problemen kann es also kommen, wenn man etwa eine Szene aus einem kopiergeschützten Programm fotografieren will. Gewöhnlich unterbricht man dann zum richtigen Zeitpunkt mit „Ctrl-Offener-Apfel-Reset“ und nachfolgen-

dem „Ctrl-Reset“, verschiebt das Bild mit dem Monitor-„Move“-Befehl in einen sicheren Teil des Speichers, bootet eine Diskette mit normalem DOS und speichert schließlich die Grafik ab. (Ctrl-Offener-Apfel-Reset zerstört übrigens je 2 Bytes je Page, wodurch das HGR-Bild geringfügig „entstellt“ wird.)

Für den, der mit dem UCSD-Pascal-System Bilder geschaffen hat, wird bei Polaroid-Palette ein Programm mitgeliefert, das diese Dateien in das DOS-3.3-Format umwandelt und damit für eine Fotografie zugänglich macht. Doppelt hochauflösende Grafik, bei der die 80-Zeichenkarte des Apple IIe eingesetzt wird, kann mit der bisher vorhandenen Software nicht verarbeitet werden.

Ist die Vorlage nun geladen, so kann man sich vor der Aufnahme ansehen, an welchen Stellen des Bildes die einzelnen der acht Apple-Farben vertreten sind. Dem geübten Palette-Benutzer bietet diese Option die Möglichkeit, das Aussehen des fertigen Fotos in etwa einzuschätzen und schon bei der ersten Aufnahme die nötigen Korrekturen vorzunehmen.

Vor der Belichtung des Films prüft das Programm diese Farbverteilung und dreht zunächst den roten Filter vor das Objektiv der Kamera. Nun wird der zunächst leere Schirm an den Stellen gefüllt, die die größte Rot-Intensität aufweisen. Der Benutzer sieht auf seinem Monitor dasselbe, nur natürlich nicht in Farbe. Einige Sekunden später füllen sich die Stellen,



Bild 3

die eine geringere Rot-Intensität aufweisen. Dieser Prozeß setzt sich fort, bis alle Teile, die einen Rot-Anteil aufweisen, erschienen sind. Der Schirm wird daraufhin gelöscht, das Filtrrad dreht sich und der Vorgang findet von neuem für die Farben Blau und Grün statt. Je nach Filmsorte dauert ein kompletter Belichtungszyklus ein bis zwei Minuten.

Wenn im voraus klar ist, welche Bilder aufgenommen werden sollen, besteht die Möglichkeit, all dies vollautomatisch ablaufen zu lassen, so daß man innerhalb von einer Viertelstunde etwa zehn fertige Dias herstellen kann.

Die oben erwähnten Rasterstreifen in der Aufnahme erscheinen bei dem Palette-System nicht mehr. Das wird dadurch erreicht, daß der Elektronenstrahl des kleinen Monitors bei jedem zweiten Durchgang um eine halbe Strahlbreite vertikal verschoben wird, so daß die horizontalen Leerstreifen, die gewöhnlich zu erkennen sind, aufgefüllt werden.

Sieht man sich bunte Apple-Grafiken auf Schwarzweiß-Schirmen an, so bemerkt man auch eine vertikale Rasterung: Alle Farben außer Schwarz und Weiß werden durch senkrechte Streifen dargestellt. Zur Erzielung unterschiedlicher Effekte kann dies bei Aufnahmen mit dem Polaroid-Palette-System beibehalten werden.

### Technische Tips

Hat man nun ein Foto vor sich liegen, so stellt man fest, daß die

Probleme, die bei normal großer Schrift auch auf Farbmonitoren auftreten, nicht behoben sind: Die vertikalen Schriftteile sind nicht weiß, sondern grün und violett. Dieser Effekt geht auf die Art der internen Farbdarstellung des Apple zurück und kann durch herkömmliche Methoden nicht behoben werden.

Das Palette-System bietet dem Benutzer jetzt aber die Möglichkeit, jede der acht Farben der Apple-Grafik in eine der 72 gespeicherten umzuwandeln. Zunächst wird festgestellt, wo eine Verbesserung nötig ist. In unserem Fall sollten die Buchstaben lesbar sein. Also gibt man dem Programm an, daß Weiß, Grün und Violett auf dem Foto in Gelb (C5) erscheinen sollen. Um auch gleich eine andere Farbkombination auszuprobieren, ersetzt man beispielsweise Orange durch Hellgrün (B6), Blau durch Rot (D6) und Schwarz durch Dunkelblau (F5). Buchstaben und Umrandung erscheinen nun einheitlich. Nachteilig ist, daß jetzt auch alle Flächen, die zuvor grün oder violett waren, gelb sind. Bei der Schaffung von Bildern muß man also darauf achten, daß sich nur die vier Farben Schwarz, Weiß, Orange und Blau später auf dem Foto unterscheiden. Mit den neuen Möglichkeiten kann man natürlich ebenso einen effektvollen hellen Hintergrund für Präsentationen schaffen.

Im allgemeinen empfiehlt es sich jedoch, am äußeren Rand sparsam mit hellen Farben umzugehen. Sonst kann es schnell geschehen, daß die leichte kissenförmige Ver-

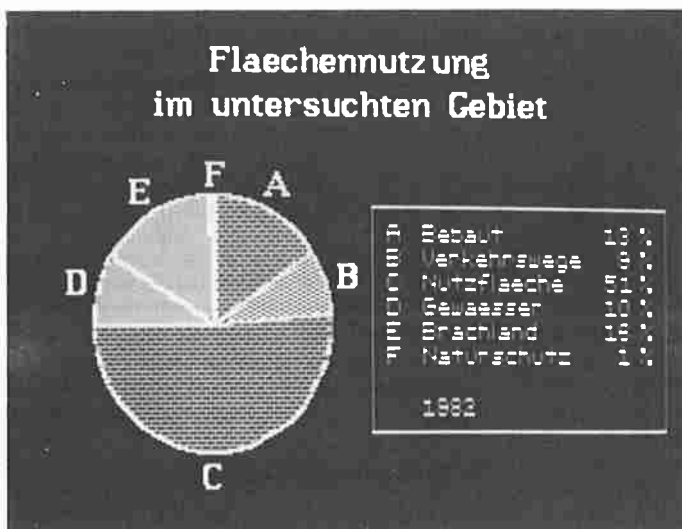


Bild 2

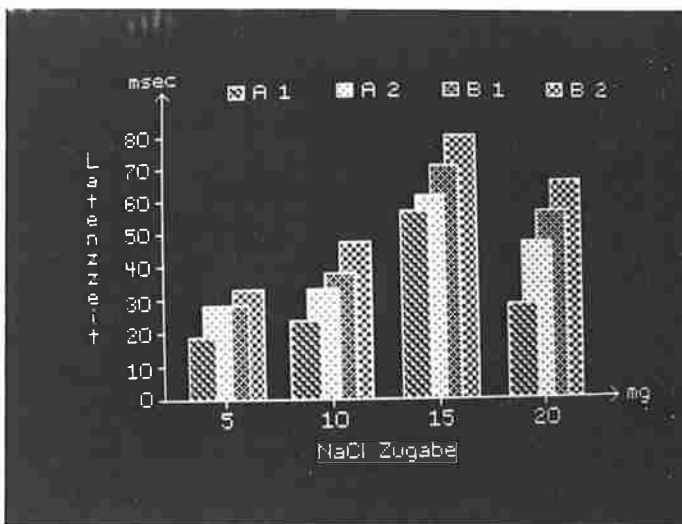


Bild 4

zeichnung des Monitor-Bildes, die sich bei der Aufnahme ergibt, ins Auge fällt.

Um dem Benutzer die Arbeit weiter zu erleichtern, liefert Polaroid bereits eine Vorschlagsliste für den Farbausch mit. Die oberste Reihe entspricht dem Apple-Standard, während die Reihen 1 bis 8 Alternativen darstellen. Es fällt auf, daß Grün, Violett und Weiß immer durch dieselbe Farbe ersetzt werden. Das geschieht, um die beschriebenen Probleme mit der Abbildung von Buchstaben und Zahlen zu verhindern.

Die Benutzerführung ist hier, ebenso wie im Rest des Programms, klar und unkompliziert. Wer mit den 72 vorhandenen Farben nicht zufrieden ist, kann selbst kreativ tätig werden und beliebige weitere Nuancen entwickeln.

**Produkt: Polaroid-Palette-System**

**Einsatz:** Foto-Dokumentation von Computer-Farbbildern für gehobene Ansprüche

**Lieferumfang:** Monitor/Rekorder, Sofortbildkamera, Kleinbildkamera mit Autowinder, Sofortdia-Entwicklungs-System, Anschlußkabel, Handbuch (80 Seiten, englisch), Programm-Diskette (kopierbar, englisch)

**Filmmaterial:** verwendbar sind diverse Sofortbild-Farbfilme, handelsübliche Dia-Filme und Polaroid-Sofortdias

**Computer:** Apple-II-Serie (jedoch keine Double-Hires-Unterstützung) und verschiedene andere PCs

**Voraussetzungen:** Computer, 1 Diskettenlaufwerk (besser 2), RS-232-C-Interface

**Preis inkl. Mwst.:** ca. DM 5600,-  
**Bezugsquelle:** Polaroid GmbH, Offenbach

**Technische Daten**

**Produkt: Polaroid-CU-5-Makro-System**

**Einsatz:** Makro-Fotografie

**Lieferumfang:** Modul-System.

**Für Sofortbilder vom 12-Zoll-Monitor:**

Kamerakörper 88-1

Linssystem 127 mm 88-5

Monitorhaube 16 · 23,3 cm 88-49

**Filmmaterial:** Polaroid Video Image Recording Land Pack Film Type 611 und weitere schwarzweiße Polaroid Land Pack Filme

**Computer:** beliebig

**Voraussetzungen:** Betriebsbereiter Schwarzweiß-Monitor

**Preis inkl. Mwst.:** in o.a. Ausstattung ca. DM 1740,-

**Abbildungen**

**Bild 1:** Motiv Heidelberg.

**Bild 2:** Kreisdiagramm Flächennutzung mit Original-Apple-Farbpalette (hier nur schwarzweiß).

**Bild 3:** Kreisdiagramm Flächennutzung mit ausgetauschten Farben, dunkler Hintergrund. Die Schrift ist nun lesbar geworden (hier nur schwarzweiß).

**Bild 4:** Technisches Balkendiagramm NaCl-Zugabe (hier nur schwarzweiß).

**ZUSATZ-KARTEN:**

V-24-Schnittstelle	199,-	Z-80-Karte	139,-
80-Zeichen-Karte m. Softswitch	236,-	16 K-Language-Karte	138,-
Centronics-Karte von Epson	für Graphik ... 210,-	für Text	145,-
Centronics-Schnittstelle für 2 Drucker gleichzeitig			129,-
Eprommer incl. Software			198,-
<b>Super-Eprommer</b>			<b>239,-</b>
belegt keinen Slot, incl. Software für 2708-27128			auf Anfrage
128-K-Karte			

**Floppy-Controller**

FDC 4 für alle Laufwerke	170,-	Bausatz wie links	159,-
Leerplatine wie oben incl. Prom u. Eprom			98,-
Druck Spooler mit 16, 32 oder 64 KB			Preis auf Anfrage

**Autopatch-Controller (Erphi Controller)**

1 x 35 bis 2 x 80 Tr.-Disk, keine Patch-Disk notwendig, Patch DOS 3.3, Diversi-DOS 2-C, 4-C (DD MOVER), Pascal 1.1, Pascal 1.2, CPM 2.2, ProDOS 1.0.1, 1.1, 1.1.1; Sie können die Laufwerke untereinander mischen

**298,-**

**Joy Stick De Luxe .. 59,- Netzteil 5A .. 149,-**

Gehäuse für 1 5/4" Slimline Laufwerk .. 39,-

Gehäuse für 2 5/4" Slimline Laufwerke mit Platz für ein Netzteil .. 159,-

Gehäuse für 2 3/4" Slimline Laufwerke mit Platz für ein Netzteil .. 79,-

IBM®-Gehäuse .. 229,-

Floppy-Kabel 34pol. für 2 Laufwerke mit Shugart-Bus .. 35,-

**Preh Commander Keyboards**

Wir bieten Ihnen die **Preh-Qualität** auch für Apple. AK 88 Spez. mit Gehäuse, Anschlußkabel, Zehner-Tastenfeld, dt. Zeichensatz, Sondertasten für Ctrl-Codes und Rechenfunktionen

**339,-**

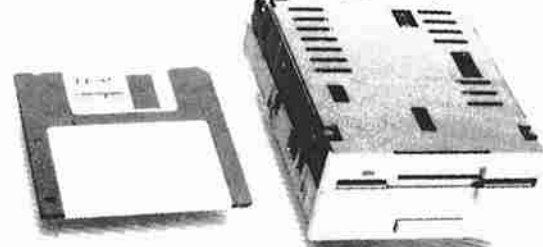
**Preh Commander Keyboard, frei programmierbar**

bis zu 10 Ebenen, pro Taste bis zu 250 Zeichen .. nur **599,-**



**TEAC 3 1/2" Laufwerk FD 35 F 535,-**

Speicherkapazität 1 MB, (formatiert 640 KB) jetzt für nur



TEAC FD 55 A 1 x 40 Track .. 445,-

TEAC FD 55 E 1 x 80 Track .. 490,-

TEAC FD 55 B 2 x 40 Track .. 515,-

TEAC FD 55 F 2 x 80 Track .. 560,-

SONY 3 1/2" Laufwerk .. nur 799,-

Apple®-kompatibles Laufwerk incl. Gehäuse + Kabel .. 599,-

**320 KB Laufwerk für IIc .. 948,-**

**148 KB Laufwerk für IIc .. 698,-**

**EPSON DRUCKER**

EPSON FX 80 .. 1670,-

EPSON FX 100 .. 2159,-

EPSON RX 80 .. 1079,-

EPSON RX 80 FT .. 1295,-

**Die Microfloppy mit Zukunft:**

Speicherkapazität: 2 x 1 MByte formatiert: 2 x 640 kByte. Anschlußfertig mit PROM-residenter Patchsoftware für CP/M 2.2, Apple DOS 3.3, Diversi-DOS 2-C, 4-C (DD MOVER), Apple Pascal 1.1, Pascal 1.2, Pro-DOS 1.0.1, 1.1, 1.1.1 zum Preis von

**1640,-**

Low Power Version .. 1740,-



**10 MB Winchester** mit Software .. **4490,-**

für DOS 3.3, CP/M 2.20, Pascal, Pro-DOS, incl. Controller und Gehäuse, bootet alle Systeme von der Winchester

**Sonderangebot**

Disketten Döb&Böd, 40 Track, SS/DD, 10er Pack .. **49,-**

Disketten Döb&Böd, 2x80 Track, SS/DD, 10er Pack .. **89,-**

Gesamt-Preisliste anfordern! Preise inklusive gesetzlicher Mehrwertsteuer.

**UEDING electronics**

Holtwiese 2  
5750 Menden 1

DFÜ 02373/66877  
Tel. 02373/63159



## Das Bildverarbeitungssystem MAGIC

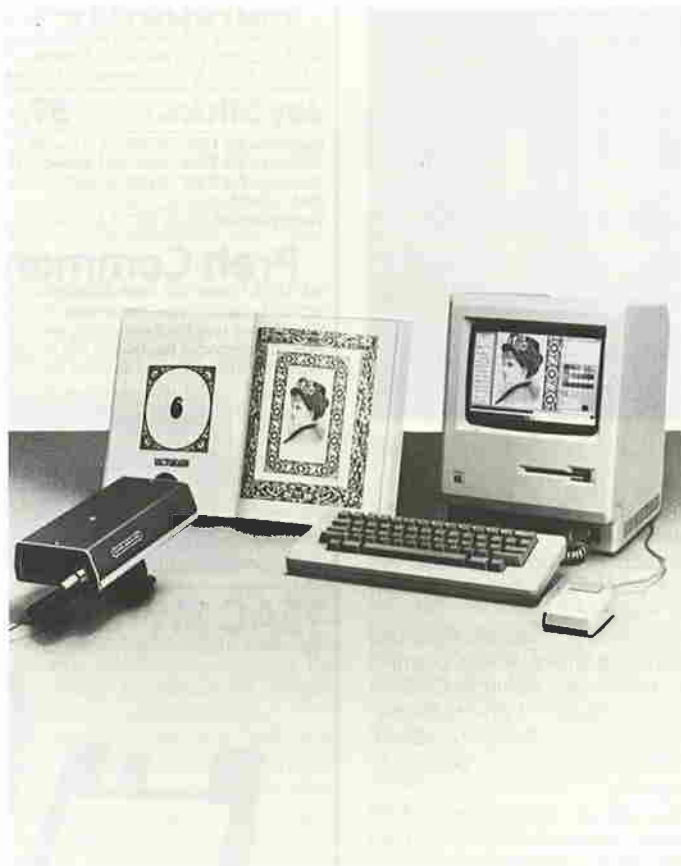
Mit dem MAGIC-Bildverarbeitungssystem für den Apple Macintosh von Heyden Datasystems können nun Bilder oder Objekte mit einer normalen Video-Kamera aufgenommen und danach auf dem Bildschirm des Macintosh wiedergegeben werden.

Das System, das aus Video-Kamera, Interface, Software und einer ausführlichen Bedienungsanleitung besteht, ermöglicht die Weiterverarbeitung der Bilder mit dem MacPaint-Programm, woraus sich ungeahnte Möglichkeiten zur grafischen Aufbereitung ergeben. Selbstverständlich können die Bilder auch in andere Anwender-Software wie MacWrite übernommen werden.

Die Handhabung ist auf Grund der Menü-Steuerung sehr einfach, wodurch auch der Computer-Laie keine Schwierigkeiten beim Ablauf der Bilderfassung hat.

Die Möglichkeit der Bild-Digitalisierung läßt sich vielfach ausnutzen:

Im wissenschaftlichen Bereich können Strukturen erfaßt und ausgewertet werden; im Bereich Grafik, Werbung und Design bietet sich die Aufnahme von Teilbildern zur gestalterischen Überarbeitung an.



## Siemens entdeckt neue Primzahl

Nachdem wir wiederholt im Peeker gezeigt haben, wie der „kleine Apple-Rechner“ mit Primzahlen umgehen kann (s. Primzahlen-Wettbewerb, Heft 1/84 und folgende Ausgaben), zeigen wir einmal, wie der „große Siemens-Rechner“ hier vorgeht und zitieren eine eingegangene Presse-notiz:

Ein Leckerbissen für Mathematiker kommt aus Hamburg: Im Rechenzentrum der dortigen Universität haben Wissenschaftler eine neue Primzahl ermittelt. Mit Hilfe eines Großcomputers vom Typ Siemens 7.882 errechneten sie die Zahl  $5 * 2 \uparrow 23473 + 1$ , die ausgeschrieben insgesamt 7067 Stellen umfaßt.

Zur Erinnerung: Primzahlen sind alle von 1 verschiedenen natürlichen Zahlen, die nur durch 1 und durch sich selbst ohne Rest teilbar sind. Das beginnt mit der 2 – übrigens die einzige gerade Primzahl – und geht weiter mit der 3, 5, 7, 11, 13, 17, usw.

Die neue Primzahl ist aber nicht nur wegen ihrer Größe bemerkenswert. Ihre Besonderheit liegt vielmehr darin, daß sie für die gigantische Fermat-Zahl  $2 \uparrow 2 \uparrow 223471 + 1$  als Teiler fungiert. Unter allen Fermat-Zahlen ist dies die weitaus größte, deren Zerlegbarkeit bislang nachgewiesen werden konnte. Fermat-Zahlen sind alle Zahlen, die sich in der Form  $2 \uparrow 2 \uparrow n + 1$  darstellen lassen.

Der in Hamburg entdeckte Teiler für die genannte Fermat-Zahl ist die viertgrößte von den derzeit bekannten Primzahlen. Die drei größten, die Primzahl  $2 \uparrow 44497 - 1$ ,  $2 \uparrow 86243 - 1$  und  $2 \uparrow 132049 - 1$  sind in den USA ermittelt worden.

Primzahlen kann man z.B. zum Verschlüsseln von Nachrichten verwenden. Dabei werden zwei sehr große Primzahlen miteinander multipliziert, was relativ einfach ist. Zum Entschlüsseln aber sind aus dem erhaltenen Produkt wieder die ursprünglichen Primzahlen zu ermitteln – und dies ist für Uneingeübte extrem rechenaufwendig.

## TRICARD – Multifunktionskarte für Apple IIe

Für alle Apple-Besitzer, die keine freien Steckplätze mehr haben, gibt es jetzt eine Multifunktionskarte, die von der Firma FAST Electronic entwickelt wurde. Diese Karte ist vollständig kompatibel zur Super-Serial-Card der Firma Apple. Darüber hinaus bietet sie eine akku-gepufferte Uhr, die mit entsprechender Treibersoftware für ProDOS und Apple Pascal betrieben werden kann. Als dritte Option ist der Betrieb eines Druckers über den Parallel-Port möglich. Die entsprechende Software und das zugehörige Centronic-Verbindungs-kabel können bezogen werden. Ein detaillierter Bericht zu dieser Karte wird demnächst erscheinen.



**Preisliste  
kostenlos!**  
Katalog DM 2,-

Unser Angebot im August

**Chinon Laufwerk**  
(Test peeker 5/85)  
DM 398,-

**Profimax III**  
(= Lazar Ilze, Test peeker 6/85)  
DM 1248,-

**D.O.S. Computersysteme**  
Am Kühnbach 42, 7170 Schwäbisch Hall 11  
Telefon (0791) 51736

# Sich regelmäßig informieren, «Peeker» abonnieren!

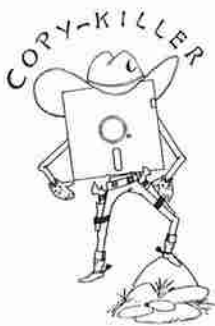
Peeker Freaks nutzen jetzt die Vorteile eines Abos:



- Regelmäßig erhalten Sie alle 12 Peeker Ausgaben des Jahres und verpassen somit keinen Bericht, kein Programm und keinen Tip
- Nutzen Sie den bequemen Einkaufsweg, denn Peeker kommt mit der Post zu Ihnen ins Haus
- Der Preis des Abonnements ist Ihr Vorteil:  
Sie zahlen DM 72,- (inkl. Porto) jährlich statt DM 6,50 Einzelpreis.  
Das ist ein Heft gratis im Jahr

## Copy-Killer jetzt in deutsch

Das bereits im Peeker 1/84 beschriebene Kopierschutzprogramm Copy-Killer ist nun in deut-



scher Version mit verbesserter Bedienungsführung erhältlich. Programme, die mit dem Copy-Killer kopiert werden, lassen sich mit den bislang bekannten Bit-Kopierprogrammen wie z.B. Nibbles Away, Locksmith etc. nicht mehr kopieren.

Beim Kopiervorgang ist wie folgt vorzugehen: Zunächst wird eine Copy-Killer Slave-Diskette angelegt. Auf diese Diskette wird die Original-DOS-Diskette kopiert. Beim Kopieren auf die Slave-Diskette kann eine Meldung eingegeben werden, die beim Booten angezeigt wird. So kann jede Diskette individuell mit einer Seriennummer versehen werden. Der Preis beträgt DM 228,-. Copy-Killer für ProDOS ist in Vorbereitung.

## Microfloppy mit 2 x 1 MByte

Die Firma Sommer GmbH bietet ein Microfloppy-Terminal MFT-A an, das die bisher üblichen 5,25-Zoll-Laufwerke ersetzt oder gemeinsam mit bereits vorhandenen Floppy-Disks betrieben werden kann. Die Einbindung in das vorhandene Betriebssystem (DOS 3.3, CP/M 2.2 oder Apple Pascal 1.1) erfolgt automatisch durch den Floppy-Disk-Controller mit PROM-residenter Patch-Software (Erphi-Controller); eine Patch-Diskette ist nicht erforderlich. Der Patch-Vorgang – das Erweitern des Betriebssystems und Einbinden der 80-Spur-Disketten – erfordert keinerlei Eingaben durch den Benutzer.

Das Doppelaufwerk in einem Metallgehäuse, das mit Kabel und

Controller geliefert wird, hat eine unformatierte Speicherkapazität von 2 MBytes oder 1,2 MBytes formatiert. Der Preis beträgt DM 1995,- (inkl. MWSt.).



## ProDOS-Debugger BUGBYTER

getestet von Dr. Jürgen B. Kehrel

Sie haben stunden- oder gar tagelang an einem Assemblerprogramm geschrieben, starten es voller Hoffnung zum ersten Mal, und zu Ihrem Entsetzen verabschiedet sich Ihr ausgeklügelter Code in das Niemandsland. Jetzt hilft nur noch eins: Mit Ruhe und einem guten Debugger (= „Entwanzer“) Schritt für Schritt den unheimlichen Spuren zu folgen.

Assemblerprogramme erzeugen einen sehr schnellen und kompakten Code, und oft ist es erst dadurch möglich, bestimmte Dinge auszuführen. Ein gewaltiger Nachteil ist aber, daß Sie keinerlei Fehlermeldungen erhalten, wenn etwas „schief geht“. Allenfalls haben Sie das Glück, die Stimme des Apple zu vernehmen, verbunden mit der Anzeige einer Adresse und der Registerinhalte. Dann wissen Sie zumindest, wo Sie gelandet, aber immer noch nicht, wie Sie dorthin gekommen sind.

Hier hilft ein Programm weiter, das seinerseits Ihr Programm in Zeitlupe ausführt und Sie über alle Vorgänge informiert. Solche „Debugger“ gibt es schon lange für den Apple, z.B. „Symbol Symon“, „The Bug“ und „Munch a Bug“. Jetzt können Sie aber mit **BUGBYTER** einen Vertreter dieser Art bekommen, der alle seine Vorgänger in den Schatten stellt.

BUGBYTER von Computer Advanced Ideas Inc. in Berkeley (Kalifornien) ist auf jeder „ProDOS Assembler Tools“-Diskette von Apple enthalten, und selbst wenn Sie den ProDOS Apple-Assembler nie benutzen werden, macht schon

der BUGBYTER den Kauf lohnend. (Früher kostete der BUGBYTER allein \$40.00)

BUGBYTER ist ein Maschinenprogramm, das sich normalerweise ab \$2000 im Speicher befindet. Was aber, wenn Ihr Code gerade hier liegt? Nun, Sie laden BUGBYTER einfach an irgendeine andere Stelle, an der knapp 7K zusammenhängend frei sind, denn BUGBYTER läuft überall, auch in der Language Card. Zusätzlich benutzt BUGBYTER den Stack im Bereich von \$0100 bis \$011F, was aber selten zu Konflikten führt.

In 6 Fenstern, deren Größe z.T. von Ihnen variiert werden kann,

**Für Apple II, IIe**

Z-80-Karte	79,-	80-Zeichen-Karte	149,-
Disk-Interface	79,-	mit Softswitch, neue Vers. m. gest. scharf. Bild	
Centronics-Interf. m. Kabel	79,-	Speech-Karte	69,-
16-K-RAM-Karte	79,-	Clock-Karte	129,-
RS-232-Karte	109,-	Super-Serial-Karte	199,-
Eprommer (4, 8, 16 K)	139,-	Komp 2E	1098,-
128-K-RAM-Karte	299,-	Apple 2E kompatible, Rechner 64 K im 2E-Design, ohne Firmware	
256-KB-RAM-Karte	598,-	80Z + 64K-Karte für 2E kompatible	99,-
Wild-Karte (knackt geschützte Programme)	99,-	Apple-Info 1,- DM (Porto)	

Händleranfragen erwünscht!

**Apple II + Kompatible** 680,-

**Komp 48**  
48 K, 6502 ohne Firmware

**Komp 64**  
64 K, 6502, Z-80, 15er-Block ohne Firmware

**880,-**

**Komp 64 S** 999,-  
wie Komp 64, jedoch mit abgesetzter Tastatur mit 188 Funktionen.

**Motherboard 48 K** 389,-  
8 Slots, alle IC's gesockelt, ohne Firmware, fertig geprüft

**Motherboard 64 K** 399,-  
wie oben, mit 6502 und Z 80, 64 K

Alle Preise inklusive Mehrwertsteuer. 6 Monate Garantie. Versand erfolgt per NN oder Vorkasse.

Hard-, Software  
Im Viertstr. 3-13  
6233 Kelkheim  
☎ (06198) 7523



sehen Sie die Register, einen Stackausschnitt, den gerade ausgeführten Code in disassemblierter Form, von Ihnen ausgewählte Speicherstellen beliebiger Lage, einen Zykluszähler und von Ihnen gewählte Stoppstellen (Breakpoints). Die unterste Zeile dient als Kommandozeile. Sie können jederzeit Code disassemblieren, neuen Code assemblieren (eine Art Mini-assembler ist eingebaut), die Register auf jeden Wert setzen oder sich Speicherauszüge in HEX oder ASCII ausgeben lassen. Sie können ferner ProDOS-Befehle (z.B. BLOAD) ausführen, in den Monitor oder nach Applesoft springen.

Um Ihr Programm zu verfolgen, lassen Sie es entweder in Einzelschritten (STEP) oder kontinu-

ierlich (TRACE) ablaufen, wobei die Geschwindigkeit über die Tastatur oder einen Joystick geregelt werden kann. Nach jedem Schritt werden alle Anzeigen aktualisiert. Unterroutinen, von denen Sie schon wissen, daß sie funktionieren, können Sie überspringen oder im Schnellgang durchheilen. Sie können sogar ganze Bereiche definieren, in denen die volle Geschwindigkeit des Apple benutzt wird, wenn es sich um zeitkritische Routinen handelt (z.B. Disk-Operationen).

Wollen Sie sehen, wie Ihr Programm die 1. oder 2. Text-, Lores- oder Hires-Seite benutzt, können sie die BUGBYTER-Anzeige zeitweise abschalten. Benutzt Ihr Programm die Tastatur, können Sie BUGBYTER mit dem Joystick

stoppen und starten und alle Tasten bis auf eine von Ihnen definierte und von BUGBYTER benötigte freigeben. Da bleibt eigentlich kein Wunsch mehr offen.

In Ihrem Programm können Sie Stoppstellen vorwählen. Transparente Stoppstellen ändern Ihren Code nicht, sondern halten die Ausführung nur an, wenn die gewählte Adresse erreicht ist. Sie können vorgeben, ob dies bereits beim ersten Erreichen der Stoppstelle geschehen soll oder erst bei einem wiederholten Male, wobei BUGBYTER dann die Zahl der tatsächlichen Durchläufe anzeigt. Auf diese Weise können Sie z.B. Schleifen austesten. Echte Stoppstellen modifizieren den Code und sind nur notwendig, wenn Sie Ihr Programm mit voller Geschwindig-

keit testen wollen. Sie brauchen sich um die überschriebenen Bytes nicht zu kümmern, denn BUGBYTER rekonstruiert sie auf Befehl.

Ich konnte Ihnen hier nur einen knappen Überblick über die Funktionen geben, deren Kombination Ihnen endlose Möglichkeiten eröffnen. Im Manual ist auf ca. 40 Seiten alles genau beschrieben. Lediglich zwei Fehler haben sich dort eingeschlichen: Auf Seite 167 muß die Adresse \$2006 und nicht \$7C06 lauten und auf Seite 169 \$2005 anstatt \$2006. Ob Sie nun ein Assembleranfänger sind und dem Apple ab und zu auf die Bytes schauen wollen oder ob Sie professionell große Programme nach Fehlern durchsuchen, mit dem BUGBYTER haben Sie dazu ein wundervolles Werkzeug.

## Ausgabe und Eingabe mit TYPETERM®

im Slot Ihres  
**APPLE II/IIe**

Das bedeutet: Computer-  
textverarbeitung von der  
Schreibmaschinentastatur!  
Steckerfertig ohne Umbau.

TYPETERM- **DM 479,-**  
Interface incl. MWSt.  
für alle BROTHER-Typenrad-  
schreibmaschinen

Paketpreis: **DM 1348,-**  
Schreibmaschine  
CE-51 mit TYPETERM



**brother**  
QUALITÄT AUS ERSTER HAND.

CE-61 mit TYPETERM ..... DM 1787,-  
EM-80 mit TYPETERM ..... DM 2087,-  
EM-100 mit TYPETERM ..... DM 3122,-  
TYPETERM-Kit für CE-50 ..... DM 468,-  
CE-25 mit TYPETERM ..... anfragen

TYPETERM – ein starkes Interface für starke Maschinen! Alle Cursor- und Ctl-Befehle. 2k ROM auf der Karte f. DOS, PRODOS, CP/M, PASCAL. Alle Features: Hoch-/Tiefstellen, autom. Unterstreichen, var. Zeichen- u. Zeilenabst., autom. Papierzuführung usw. Ausführl. Handbuch vorab: 10,- DM auf Konto 14770-306 PGiroA Han (Anrechnung).

TYPETERM ein Produkt von

**interkom** Kock & Mreches GmbH  
Postf., 3004 Isernhagen 4  
Telefon 05139-87393  
**electronic**

## Beagle Graphics

getestet von Rolf W. Becker

Beagle Graphics ist ein hervorragendes Zeichenhilfsmittel in der doppelt hochauflösenden Grafik (560 \* 192 Punkte), wobei 16 Grundfarben und 240 Mischfarben (in schwarzweiß entspricht dies verschiedenen Schraffuren) zur Verfügung stehen. Die Doppelt-Lores-Grafik (80 \* 40 und 80 \* 48 Punkte) wird ebenfalls unterstützt. Gezeichnet werden kann mit der Apple-Mouse, dem Apple Graphics Tablet, dem Joystick bzw. Paddle oder dem Koalpad und schließlich auch mit der Tastatur. Die Diskette enthält sowohl eine DOS-3.3- als auch eine ProDOS-Version.

Nach dem Booten wählt man mit „<Ctrl-D> RUN DOUBLE.PLOT“ das Modul zum Zeichnen und bestimmt danach eine der obengenannten Zeichenhilfen. Das darauffolgende Haupt-Menü bietet folgende Optionen:

- **Box** zeichnet Rechtecke,
- **Circle** zeichnet Kreise und Ellipsen,
- **Draw** stellt einen Bleistift zum Freihandzeichnen zur Verfügung,
- **Line** zieht eine Linie mit dem Bleistift,
- **Edit** bearbeitet einen festzulegenden Teil,

- **Fill** füllt umschlossene Flächen mit Mustern oder Farben,
  - **Paint** ermöglicht das Zeichnen mit verschiedenen Pinselstärken,
  - **Text** schreibt in 21 verschiedenen Schriftarten,
  - **Mode** wählt zwischen den Darstellungsgrößen,
  - **Set Color** legt die Vorder- und Hintergrundfarbe fest,
  - **X** macht den Bildschirm in wählbarer Farbe frei,
  - **Quit** verläßt das Programm.
- Mit der Leertaste kann ein kleiner Teil des Bildes in Einzelpunkten bearbeitet werden.

Mit dem Edit-Befehl wird zunächst ein Gebiet der Zeichnung festgelegt, das dann bearbeitet werden kann. Dabei besteht die Möglichkeit, Gebiete zu löschen, zu kopieren oder zu verschieben. Weiterhin können die festgelegten Teile invertiert und in horizontaler und vertikaler Richtung verdreht werden. Mit der ESC-Taste kehrt man in das Haupt-Menü zurück.

Mit <Ctrl-F> wird der Fill-Befehl angesprochen, der alle Möglichkeiten von Farben und Strukturen eröffnet. Damit ist Beagle Graphics, soweit ich informiert bin, das Programm, mit dem die größte Palette auf dem Apple erreicht werden kann. Vor dem Füllen empfiehlt es sich, das Bild auf Diskette zu speichern, um den Vorgang zu wiederholen, falls etwas schiefgeht.

Der Paint-Befehl ermöglicht im Gegensatz zu MousePaint, das nur schwarze Pinselstriche erlaubt, auch andere Muster. Es stehen 16 Pinselarten zur Verfügung, wobei mit einem Pinselstrich in der Bildschirmfarbe auch ein Radiergummi benutzt werden kann.

Mit dem Text-Befehl kann in einer der 21 verschiedenen Schriftarten (auch Russisch und Griechisch), die auf Diskette mitgeliefert werden, geschrieben werden. Darüber hinaus ermöglicht das Font-Editor-Programm die Erstellung eigener Zeichensätze.

Mit <Ctrl-D> wählt man Diskettenbefehle. Doppelt-Hires-Bilder werden automatisch in zwei Files gespeichert; der zweite mit dem Zusatz „.AUX“. Zusätzlich befindet sich eine „SLIDE.SHOW“ auf Diskette, mit der man seine Doppelt-Hires-Bilder wie in einer Diashow auf dem Bildschirm ablaufen lassen kann.

Beagle Graphics beinhaltet 33 neue Applesoft-Befehle, die alle sehr gut und ohne Probleme in eigene Programme einfügbar sind und auch alle ausgezeichnet funktionieren. Außerdem gibt es einige interessante und nützliche Utilities wie z.B. die Umwandlung von eigenen Hires- in Doppelt-Hires-Bilder (auch für Lores). Besonders nützlich ist „DOUBLE.

SCRUNCH". Damit werden die beiden Bild-Files auf Diskette zu einer kleineren Datei zusammengefaßt. Man kann so mehr Bilder für die „SLIDE.SHOW“ auf einer Diskette speichern. Applesoft-Graphikprogramme werden mit der „HGR.TO.DHGR“- bzw. „GR.TO.DGR“-Utilitie automatisch umgewandelt, haben dann also den neuen Befehlsatz.

Ich habe das Programm mit der Apple-Mouse, dem Joystick und der Tastatur getestet. Mit der Tastatur ist das Zeichnen mühselig, führt aber zu guten Ergebnissen. Mit dem Joystick ist die Handhabung sehr einfach, doch meiner Ansicht nach auch sehr ungenau, vor allem, wenn man exakt positionieren will. Am besten konnte ich mit der Apple-Mouse zeichnen.

Zwar gibt es hier nicht die Möglichkeit, wie bei MousePaint alle Verarbeitungsmöglichkeiten über Fenster anzuklicken – die Befehle werden über Einzeltasten wie „E“ für Edit eingegeben –, doch die Umstellung fällt nicht schwer. Die Resultate sind hervorragend. Jeder Punkt der Zeichnung ist genau erreichbar; kleinste Details können exakt eingezeichnet werden. Leider war die entsprechende Treiber-Software für die Doppelt-Hires noch nicht lieferbar (es wird „Triple Dump“ von Beagle Bros Inc angeboten). Deshalb konnte ich noch keine Bilder drucken. Hardware-Voraussetzungen: Apple IIc oder IIe mit erweiterter 80-Zeichen-Karte. Erhältlich ist dieses Programm für ca. DM 240,- bei einschlägigen Importeuren.

### Finanzbuchhaltung

für CP/M-80-Systeme, z. B. **Apple, Proteus, TRS-80;**  
für Commodore **CBM 8032/4032** mit Diskettenlaufwerk.

### Anlagenbuchhaltung

(unter anderem für Abschreibungsberechnung)  
für CP/M-80-Systeme, z. B. **Apple, Proteus, TRS-80.**  
Die CP/M-Programme sind auch unter MP/M lauffähig.

Bitte geben Sie bei Anfragen Ihren Computertyp und die Ausbaustufe an.

**Programmierbüro Kurt Kastner, Nikolausstr. 3**  
**7500 Karlsruhe-Rüppurr · Telefon 07 21 / 88 42 90**

### Inserentenverzeichnis peeker 8/85

	Seite
aaa electronic gmbh, Freiburg . . . . .	25
ccp-datentechnik, Hamburg . . . . .	59
U. Dobbertin, Brühl . . . . .	35
D.O.S. Computersysteme, Schwäbisch Hall . . . . .	74
HIB, Nürnberg . . . . .	78
IBS Computertechnik, Bielefeld . . . . .	U4
Interkom electronic, Isernhagen . . . . .	77
Intus, Waldshut-Tiengen . . . . .	66
Jeschke, Kelkheim . . . . .	76
Kurt Kastner, Karlsruhe-Rüppurr . . . . .	78
E.-W. Meyer, Frohnhausen . . . . .	23
Micromint Computer GmbH, Erkrath . . . . .	59
U. Mohwinkel Electronic, Leverkusen . . . . .	35
Pandabooks, Berlin . . . . .	21
Pandasoft, Berlin . . . . .	11
Summagraphics, München . . . . .	47
Tombstone-Micro, Berlin . . . . .	35
Ueding electronics, Menden . . . . .	73



**AKUSTIK-KOPPLER - Dataphon s21d**  
300 Baud Modem, nach CCITT V.21 Standard,  
m. FTZ-Nr. 18.13.1917.00, Gebühren- und  
anmeldefrei, V24/RS-232 Standard-Schnittst. nur DM 298,00

**TELEKOMMUNIKATIONS - KOMPLETT - PAKET**  
geeignet für Apple //+ und Apple //e:  
1 Dataphon s21d,  
1 Anschlußkabel: V.24 zum Apple II-Game-I/O,  
1 Terminalprogramm: "HIB Modem-Transfer" nur DM 398,00

**Chinon-Laufwerk (Testbericht in Peeker 5/85)**  
für Apple //+ und Apple //e anschluf. im Gehäuse DM 498,00  
w.o. jedoch für Apple //c DM 569,00

**TOSHIBA Spitzenlaufwerke zum Superpreis!**  
ND 06-D, 2 x 80 Track, 640 K-Byte formatiert DM 549,00

**DISK-DOPPEL-STATION (APPLE //+, APPLE //e)**  
2 x ND 06-D im Geh. + Auto-Patchcontr., 1,2 MB DM 1698,00

**AUTO-PATCH-CONTROLLER** DM 298,00

**IC-Test-Karte (Testet ca. 500 verschiedene IC's)** DM 398,00

**BROTHER-Matrixdrucker, die Super-Drucker!**  
M-1009 (Matrixdrucker, RS-232 + Centronics) DM 698,00  
M-1009 anschlufertig an:  
Apple //c (mit Drucker-Kabel) DM 798,00  
Apple //e (mit Graphik-Interface und Kabel) DM 898,00

Alle Preise inklusive der gesetzlichen Mehrwertsteuer.  
Berechnung der Versandkosten erfolgt nach Entfernung und Gewicht.  
Fordern Sie noch heute unsere **Gratispreisliste** an Wiederverkäufer bitte nur schriftlich anfragen (Kopie der Gewerbeanmeldung beilegen!).

**hib** HIB Computerladen  
Auß. Bayreuther Str. 72 - Telefon: 0911 / 515 939  
Postfach 21 01 25 - Telex: 17 - 911 8253  
8500 Nürnberg 21 - Teletex: 2526 - 911 82 53

Einem Teil dieser Ausgabe liegt ein Prospekt der Firma Interdata GmbH, Singen bei.  
Wir bitten unsere Leser um Beachtung.

**Vergriffene Peeker-Hefte**

**Abonnenten können vergriffene Peeker-Hefte als Heft-Kopien erwerben. Inlandspreis DM 10,- inkl. Versandkosten. Auslandspreis DM 12,- inkl. Versandkosten (Luftpostzusendung extra).**

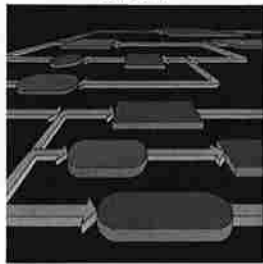
**Z. Zt. sind vergriffen Heft 1/1984 und Heft 1-2/1985**

APPLE II  
PASCAL  
Betriebssystem



te-wi

APPLE II  
PASCAL  
Sprache



te-wi

# APPLE II PASCAL

## Betriebssystem und Sprache

**Erstes deutsches Referenzwerk** sämtlicher Befehle und Systemroutinen von Apple II Pascal – mit Addendum einschließlich Version Pascal 1.2!

**Gültig für Apple II, II Plus, IIe** einschließlich der 128K/80 Zeichen-Konfiguration.

**Betriebssystem** kommentiert ausführlich und in Deutsch Funktion und Benutzung der fast 60 Systemroutinen des Apple II Pascal Betriebssystems.

**Sprache** ist das vollständige, deutsche Referenzwerk der „Apple Pascal“-Programmiersprache mit u. a. Informationen über professionelle Pascal-Programmierung, Turtlegraphics, Programmibibliothek etc.

**In Vorbereitung: Addendum Pascal 1.2**, ein Zusatz zum Buch „Betriebssystem“ für 1.2-Benutzer in Deutsch.

„Nach Unterlagen von Apple Deutschland hergestellt“

Apple II Betriebssystem,  
272 Seiten, DM 49,-

Apple II Sprache,  
216 Seiten, DM 39,-

Pascal 1.2 Addendum, 112 Seiten,  
DM 36,-

te-wi Verlag GmbH  
Theo-Prosel-Weg 1  
8000 München 40

te-wi

## Weiterführende Literatur...



**Das APPLE II - Handbuch**

(L. Poole)  
Erst mit Hilfe dieses Leitfadens werden Sie Ihren Apple II erfolgreich einsetzen, denn Text und Bildmaterial gehen weit über das hinaus, was herstellereitig an Literatur angeboten wird.

Neu überarbeitet und jetzt um die spezifischen Eigenheiten der Modelle **IIe** und **IIc** erweitert. 500 Seiten, Softcover, DM 66,-

NEU



**LOGO - Jeder kann programmieren**

(Daniel Watt)  
Buch des Jahres in den USA. Für die Computer APPLE II, C-64, IBM PC, ATARI bis 520 ST, TI-99 und CPC 464/664. Hochwertiges Textbuch für Logo-Kurse für zu Hause und im Lehrbereich. 384 Seiten, A4, DM 59,-



**APPLE II PASCAL - Eine praktische Anleitung**

(A. Luehrmann, H. Peckham)  
Unentbehrlich für alle, die die Programmiersprache PASCAL lernen wollen und Zugang zu einem Apple-Computer haben. 544 Seiten, Softcover, DM 59,-



**APPLE II - Bewegte 3D-Graphik**

(Phil Cohen)  
Selbstentworfenene Graphiken und Diagramme – animiert oder als Standbilder – eben oder räumlich: alle erforderlichen BASIC-Programme mit Erklärung finden Sie in diesem Buch. 200 Seiten, Softcover, DM 49,-



**Computer für Kinder**

(Sally Greenwood Larson)  
Ein Buch für Kinder, ihre Lehrer und Eltern.

„Computer für Kinder“ richtet sich an Kinder im Alter von 8 bis 13 Jahren, für deren Interesse an Computern dieses Buch bewußt geschrieben wurde.

Unterhaltsam und leicht verständlich, A4 quer. Fadenheftung. DM 29.80



**Apple Maschinensprache**

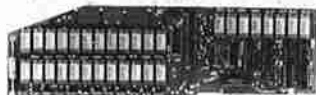
Für BASIC-Programmierer der einfachste Zugang zur Muttersprache des Apple. Wesentlich schnellere Maschinenprogramme, direkte Manipulation des Mikroprozessors 6502 im Apple – als Brücke dorthin benötigt dieses Buch nur die drei BASIC-Befehle, POKE, CALL, PEEK. D. Inman/K. Inman. DM 49,-

**NEU: REPARATURANLEITUNG  
COMPUTER: APPLE II, II PLUS  
DM 29,80**

Noch im Programm:  
6502 - Programmieren in Assembler DM 59,-  
VisiCalc, 50 Programme auf Diskette, DM 79,-

In Vorbereitung:  
Macintosh Programmier-Handbuch DM 59,-





AP 13 und AP 17  
RAM-Karten zum Einsatz als Pseudodisk unter CP/M, USCD und APPLE-DOS. Speichergröße von 64 kByte bis 256 kByte.  
Bestell-Nr.: A 1013 a-b  
A 1017 a-d



AP 33  
RAMDISK der neuen Generation. Für besonders speicherintensive Arbeiten ist der Ausbau in Stufen von 64 kByte bis 1MByte möglich.  
Bestell-Nr. A 1033

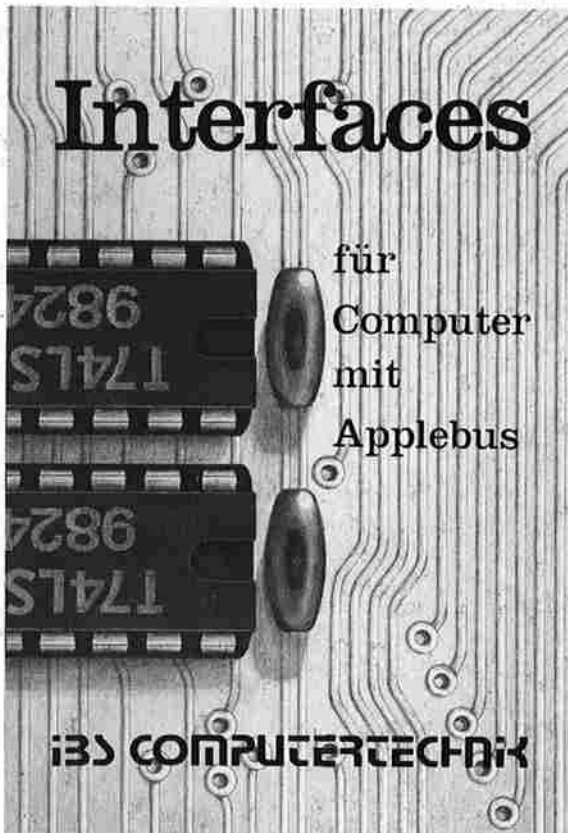


AP 14  
Floppy-Controller für alle Anwendungsfälle. 10 Laufwerke können gleichzeitig angeschlossen werden. 4 x 8" DSDD, 4 x 5 1/4" DSDD und zwei Apple-Standardlaufwerke. Maximal ca. 10MByte im Direktzugriff.  
Bestell-Nr.: A 1014



**NEU!** jetzt 512 k-RAM

AP 20  
INTEMEX mit 68 000 CPU und 128 k-RAM. Diese Karte macht aus Ihrem Rechner mit „Applebus“ einen echten 16 bit-Rechner. Eine Zusatzkarte (AP 26) ermöglicht einen Arbeitsspeicher bis zu einem MByte und an Software gibt es einiges. Z.B. stehen drei Betriebssysteme und die wichtigsten Hochsprachen zur Verfügung.  
Bestell-Nr. A 1020



AP 19  
12-Kanal AD-DA-Wandler mit 12 bit Auflösung und 25  $\mu$  sec Wandlungszeit. Eingangsspannung  $\pm 10$  V. Ein schneller Wandler für extrem schnelle Anwendungen.  
Bestell-Nr.: A 1019



**NEU!** 8 MHz Takt

AP 22  
INTEMEX mit Z 80 B-CPU und 64 k-RAM. Wenn Sie einmal diese Karte in Aktion gesehen haben, werden Sie auch feststellen: „Geschwindigkeit ist keine Hexerei, man braucht nur die AP 22“. Mit dieser Karte wird Ihr APPLE II zum z.Z. schnellsten CP/M-Computer, und in Verbindung mit dem SPACE 84 erhalten Sie Computerleistung, die wirklich einmalig ist. Wir vermitteln gerne eine Vorführung.  
Bestell-Nr. A 1022

**NEU!!!**

Das Interface-Buch von IBS, ein Buch für Alle, die Ihren APPLE II oder Kompatiblen optimal nutzen wollen. Detaillierte Schaltpläne, Bauteilelisten und Benutzungshinweise zu allen IBS-Interfaces finden Sie jetzt in einem Buch vereint. Ausführliche Abhandlungen über Spezialschaltungen, über Anwendungsmöglichkeiten, über neue Softwarewelten aber auch über die Grenzen des APPLE II-Systems bestimmen den Wert dieses Buches.

Für nur DM 8,00 erhalten Sie dieses Buch ab sofort bei Ihrem Computerfachhändler oder für DM 8,00 + DM 2,00 Versandkosten bei IBS COMPUTERVERTRIEB.

**IBS**  
**COMPUTERTECHNIK**

Olper Straße 10 · 4800 Bielefeld 14 · Tel.: 0521/444032 · W. Germany  
1011 Rose Marie Lane 16 · Stockton CA 95207 · Tel. 209/473-7473 USA